Case 1 $t = A[m]$

As the function works this will return m

Case 2 t does not exist in $A[m]$

As the array is sorted when $A[m]$ should be equal to t instead it will find $A[m] < t$ thus the function will be called oncemore making right > left returning -1

Case 3: $t < A[m]$

We know, as the array is sorted, that t, if it exists, must lie between $A[left]$ [redacted] and $A[m-1]$. If the recursive call works correctly than so must this one. So $n = (m-1) - left + 1$ or $n = (\lfloor \frac{left+right}{2} \rfloor - 1) - left + 1$. So if left+right is even then $n = \frac{left+right}{2} - \frac{2left}{2} - 1 + 1 = \frac{right-left}{2} + 1 - 1$. When odd $n = \frac{left+right-1}{2} - \frac{2left}{2} - 1 + 1 = \frac{right-left}{2} + 1 - 1$ thus $\frac{right-left-1}{2} + 1 - 1 < \frac{right-left}{2} + 1 - 1 \leq$ ▽

▽ $\leq right - left + 1$ so $\frac{right-left}{2} + 1 - 1 \leq k$ [redacted]. thus the binary search works after this [redacted] recursive call in the range of 1 to k and must be correct by Hyp (works on $k+1$ interval)

Case 4 $t > A[m]$

We know as the array is sorted that t must, if it exists, lie between $A[m+1]$ and $A[right]$. If the recursive call works so must this one. So $n = right - (m+1) + 1$. So if left+right is even then $n = \frac{2right}{2} - \frac{right+left}{2} - 1 + 1 = \frac{right-left}{2}$. When odd $n = \frac{2right}{2} - \frac{right-left+1}{2} - 1 + 1 = \frac{right-left+1}{2}$ as $\frac{right-left}{2} \leq right - left + 1$ and $\frac{right-left+1}{2} \leq right - left + 1$ then as both are $\leq k$ then the recursive call is made on an interval between 0 and k and is correct by Hyp thus working on interval $k+1$.

Thus as the inductive step works in all cases, we can see ~~exp~~ search works in all cases. ■