Consider the following computational problem:

**First Nonzero Entry in Part of an Array**

*Precondition:* An integer array $A$, with positive length $n$, and integers $low$ and $high$, such that $0 \leq low \leq high \leq n - 1$, are given as input.

*Postcondition:* If at least one of

\[ A[low], A[low+1], \ldots, A[high] \]

is nonzero, then $A[i]$ is returned as output, where $i$ is the smallest integer such that $low \leq i \leq high$ and $A[i] \neq 0$. The value 0 is returned otherwise.

Consider, as well, the following recursive algorithm:

```java
integer firstNonZero ( integer[] A, integer low, integer high ) {
1. if (low == high) {
2.  return A[low]
} else {
3.  integer mid := floor((low + high)/2)
4.  integer firstChoice := firstNonZero(A, low, mid)
5.  if (firstChoice != 0) {
6.    return firstChoice
} else {
7.    return firstNonZero(A, mid+1, high)
}
}
```
1. Give traces of executions and the recursion tree for the execution of this algorithm for an input matrix $A$ with length $8$ such that, for $0 \leq i \leq 7$,

$$A[i] = \begin{cases} 1 & \text{if } i = 3, \\ 0 & \text{otherwise}, \end{cases}$$

and for inputs $\text{low} = 0$ and $\text{high} = 7$. The information you discover and document, when doing this, might help you to solve the problems that follow.

2. Give a bound function for this recursive algorithm and show that it is correct.

3. Prove that this algorithm correctly solves the “First Nonzero Entry in Part of an Array” problem — writing this proof as carefully and as well as you can.

4. Give a set of assertions that can be used to document the correctness of this algorithm as inline code.