The binary tree corresponding to a heap has \( n \) nodes, a depth of \( h \), and \( n_h \) leaves in its last level. It is known that \( n \) is odd.

1. Is \( n_h \) even, odd, or it is not possible to determine? Explain your answer.

2. How many internal nodes are there in level \( h - 1 \)? Express your answer only in terms of \( n \), \( h \) and \( n_h \) (you do not have to use all of them).

Answers:

1. \( n = \sum_{i=0}^{h-1} 2^i + n_h \).

\[ \sum_{i=0}^{h-1} 2^i \] is odd. For \( n \) to be odd, \( n_h \) must be even.

2. \( n_h \) leaves in level \( h \) have \( n_h/2 \) parents in level \( h - 1 \). These parents are internal nodes. The remaining nodes in level \( h - 1 \) are leaves. Therefore, there are \( n_h/2 \) internal nodes in level \( h - 1 \).
The binary tree corresponding to a heap has $n$ nodes, a depth of $h$, and $n_h$ leaves in its last level. It is known that $n_h$ is even.

1. Is $n$ even, odd, or it is not possible to determine? Explain your answer.

2. How many leaves are there in level $h - 1$? Express your answer only in terms of $n$, $h$ and $n_h$ (you do not have to use all of them).

1. $n = \sum_{i=0}^{h-1} 2^i + n_h$. 

   $\sum_{i=0}^{h-1} 2^i$ is odd. If $n_h$ is even, $n$ must be odd.

2. $n_h$ leaves in level $h$ have $n_h/2$ parents in level $h - 1$. These parents are internal nodes. The remaining nodes in level $h - 1$ are leaves. The total number of nodes in level $h - 1$ is $2^{h-1}$. The number of leaves is $2^{h-1} - n_h/2$. 