Consider a Dictionary whose keys belong to a totally ordered set, and whose values are real numbers. We would like to implement an additional operation: \texttt{partialsum}(k, k') which sums all values whose key $\ell$ satisfies $k \leq \ell < k'$.

We can implement this Dictionary using a balanced binary search tree, and implement \texttt{partialsum} by first searching for $k$ then calling \texttt{successor} until we reach a key $\ell \geq k'$ or we run out of keys. (The \texttt{successor} function, when applied to a node in the tree whose key is $k$, returns the node with the smallest key that is $> k$, if one exists.)

We can analyse the cost of \texttt{partialsum} by treating it as a sequence of operations: one search, then one or more calls to \texttt{successor}. We can analyse the cost of this sequence using the potential method.

\begin{itemize}
  \item \textbf{(a)} In the tree shown above, label nodes by the order in which they are visited by successive calls to \texttt{successor}, starting from the shaded node. \hspace{1cm} \text{[2 marks]} \hspace{1cm}
  \item \textbf{(b)} Give pseudocode for the \texttt{successor} function. Show that the worst-case cost of \texttt{successor} is $\Omega(\log n)$, where $n$ is the number of items in the tree. \hspace{1cm} \text{[5 marks]} \hspace{1cm}
  \item \textbf{(c)} Consider the function
    \[ \Phi(k) = 2r_k + D - d_k \]
    where $D$ is the depth of the tree, $r_k$ is the number of right-child steps on a path from root to the node with key $k$, and $d_k$ is depth of that node. Augment $\Phi$ by defining its value at an ‘initial empty’ state, which you should define. Explain why your augmented function is a potential function. \hspace{1cm} \text{[3 marks]} \hspace{1cm}
  \item \textbf{(d)} Show that \texttt{partialsum} is $O(m + \log n)$, where $n$ is the number of items in the tree and $m$ is the number of calls to \texttt{successor}. Explain your reasoning. \hspace{1cm} \text{[10 marks]} \hspace{1cm}
\end{itemize}