Range query in treaps

$\textbf{Treap-Range-Search}(T, 28, 47, 36) : \ [x_1, x_2] = [28, 47], \ [y_1, y_2] = [36, \infty) $

- BST-Range-search$(x_1, x_2)$ to get boundary and allocation nodes.
Range query in treaps

\textit{Treap-Range-Search}(T, 28, 47, 36) : \ [x_1, x_2] = [28, 47], \ [y_1, y_2] = [36, \infty) \\

- \textbf{BST-Range-search}(x_1, x_2) \textit{ to get boundary and allocation nodes.}
- \textbf{Boundary-nodes}: Explicitly test whether in \textit{x-range} and \textit{y-range}. 

\begin{itemize}
  \item BST-Range-search(x_1, x_2) \textit{ to get boundary and allocation nodes.}
  \item Boundary-nodes: Explicitly test whether in \textit{x-range} and \textit{y-range}. 
\end{itemize}
Range query in treaps

**Treap-Range-Search** \((T, 28, 47, 36) : \ [x_1, x_2] = [28, 47], \ [y_1, y_2] = [36, \infty) \)

- BST-Range-search\((x_1, x_2)\) to get boundary and allocation nodes.
- Boundary-nodes: Explicitly test whether in \(x\)-range and \(y\)-range.
- Allocation-nodes: If \(y \geq y_1\), report and recurse in children.
Range query in treaps

**Treap-Range-Search** $(T, 28, 47, 36) : \quad [x_1, x_2] = [28, 47], \; [y_1, y_2] = [36, \infty)$

- **BST-Range-search**$(x_1, x_2)$ to get boundary and allocation nodes.
- **Boundary-nodes:** Explicitly test whether in $x$-range and $y$-range.
- **Allocation-nodes:** If $y \geq y_1$, report and recurse in children.
Root stores:
- Point $\bar{p}$ that maximizes $y$-coordinate
- Median $\hat{x}$ of the remaining points.

Items in left and right subtrees split by $\hat{x}$

Range-search is very similar to treaps, use $\hat{x}$ to find path

Height is $O(\log n)$