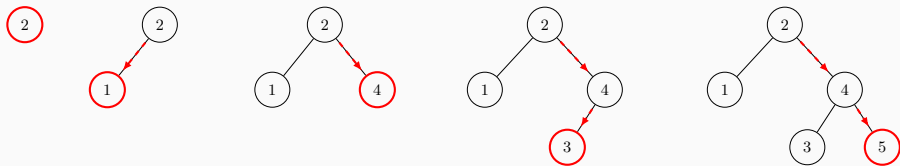


- **Problem:** Given a number  $n$  ( $1 \leq n \leq 2 \cdot 10^5$ ), design an ordering of the numbers  $1, 2, \dots, n$  to insert elements into a binary search tree (BST), such that the BST satisfies the AVL property. Find the lexicographically smallest such ordering.

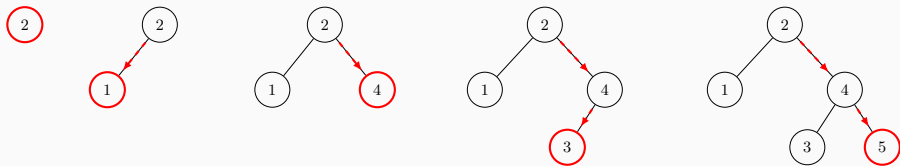


**Figure 1:** Inserting the values in order (2, 1, 4, 3, 5) in a BST.

## B: Building Beaver

Problem author: Jeroen Op de Beek

- **Problem:** Given a number  $n$  ( $1 \leq n \leq 2 \cdot 10^5$ ), design an ordering of the numbers  $1, 2, \dots, n$  to insert elements into a binary search tree (BST), such that the BST satisfies the AVL property. Find the lexicographically smallest such ordering.



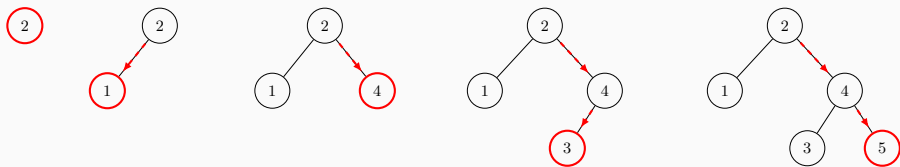
**Figure 1:** Inserting the values in order (2, 1, 4, 3, 5) in a BST.

- **Fun fact:** AVL rebalancing shows solution always exists. It does not help finding the smallest such ordering.

## B: Building Beaver

Problem author: Jeroen Op de Beek

- **Problem:** Given a number  $n$  ( $1 \leq n \leq 2 \cdot 10^5$ ), design an ordering of the numbers  $1, 2, \dots, n$  to insert elements into a binary search tree (BST), such that the BST satisfies the AVL property. Find the lexicographically smallest such ordering.



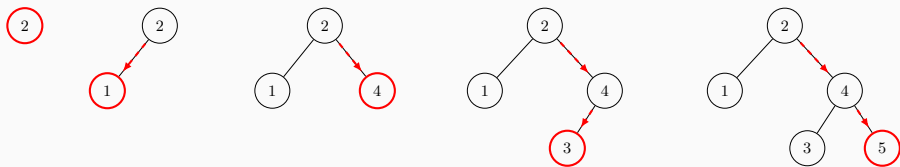
**Figure 1:** Inserting the values in order (2, 1, 4, 3, 5) in a BST.

- **Fun fact:** AVL rebalancing shows solution always exists. It does not help finding the smallest such ordering.
- **Observation:** The first number  $p_1$  becomes the root of the BST.

## B: Building Beaver

Problem author: Jeroen Op de Beek

- **Problem:** Given a number  $n$  ( $1 \leq n \leq 2 \cdot 10^5$ ), design an ordering of the numbers  $1, 2, \dots, n$  to insert elements into a binary search tree (BST), such that the BST satisfies the AVL property. Find the lexicographically smallest such ordering.



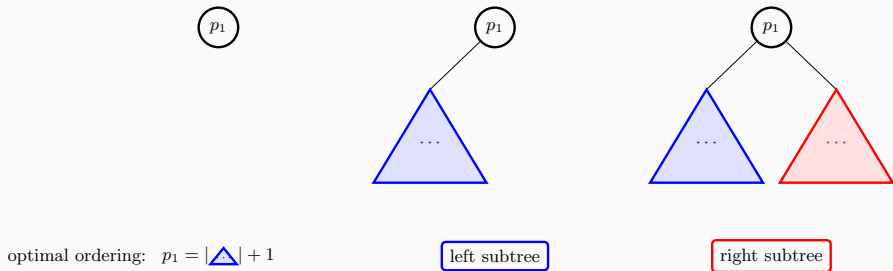
**Figure 1:** Inserting the values in order (2, 1, 4, 3, 5) in a BST.

- **Fun fact:** AVL rebalancing shows solution always exists. It does not help finding the smallest such ordering.
- **Observation:** The first number  $p_1$  becomes the root of the BST.
- All  $p_i < p_1$  form the left subtree. All  $p_i > p_1$  form the right subtree.

# B: Building Beaver

Problem author: Jeroen Op de Beek

- **Observation:** Optimal permutation will look like:

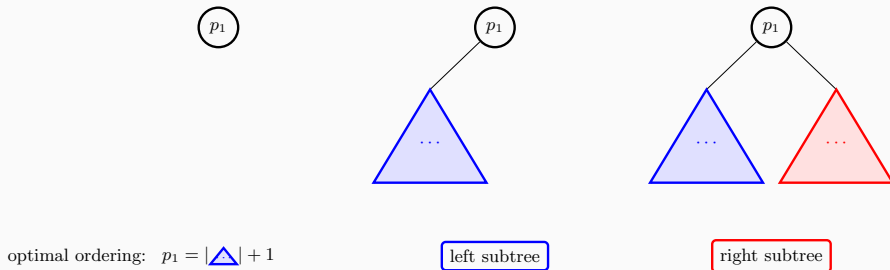


**Figure 2:** Lexicographically smallest AVL ordering.

# B: Building Beaver

Problem author: Jeroen Op de Beek

- **Observation:** Optimal permutation will look like:



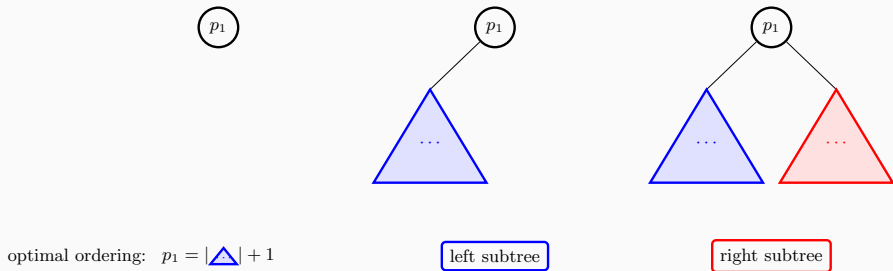
**Figure 2:** Lexicographically smallest AVL ordering.

- **Observation:** Left and right subtree are independent subproblems, can be recursively solved.

# B: Building Beaver

Problem author: Jeroen Op de Beek

- **Observation:** Optimal permutation will look like:



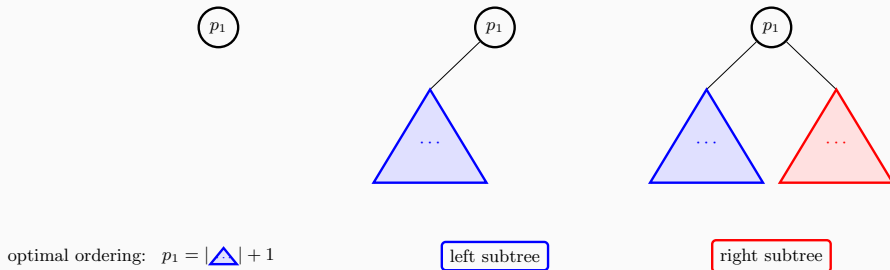
**Figure 2:** Lexicographically smallest AVL ordering.

- **Observation:** Left and right subtree are independent subproblems, can be recursively solved.
- **Solution:** Make a recursive function  $\text{avltree}(n, h_{\text{LO}}, h_{\text{HI}})$

# B: Building Beaver

Problem author: Jeroen Op de Beek

- **Observation:** Optimal permutation will look like:



**Figure 2:** Lexicographically smallest AVL ordering.

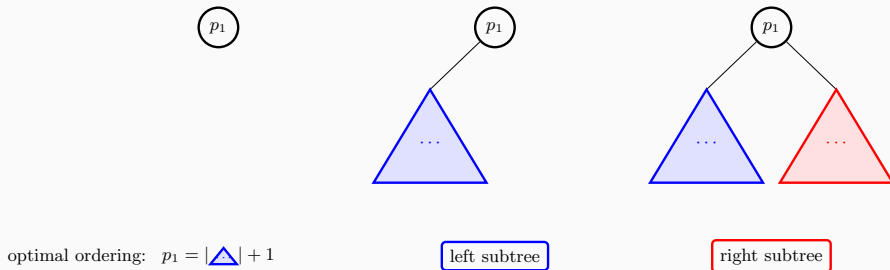
- **Observation:** Left and right subtree are independent subproblems, can be recursively solved.
- **Solution:** Make a recursive function  $\text{avltree}(n, h_{\text{LO}}, h_{\text{HI}})$
- Will construct lexicographically smallest ordering for an AVL tree with  $h_{\text{LO}} \leq \text{height} \leq h_{\text{HI}}$ .



# B: Building Beaver

Problem author: Jeroen Op de Beek

- **Observation:** Optimal permutation will look like:



**Figure 2:** Lexicographically smallest AVL ordering.

- **Observation:** Left and right subtree are independent subproblems, can be recursively solved.
- **Solution:** Make a recursive function  $\text{avltree}(n, h_{\text{LO}}, h_{\text{HI}})$
- Will construct lexicographically smallest ordering for an AVL tree with  $h_{\text{LO}} \leq \text{height} \leq h_{\text{HI}}$ .
- Want to minimize  $p_1$  first, so want to minimize left subtree size

# B: Building Beaver

Problem author: Jeroen Op de Beek



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:

# B: Building Beaver

Problem author: Jeroen Op de Beek



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:
- $S_{\min}(h) = S_{\min}(h - 2) + S_{\min}(h - 1) + 1$ ,  $S_{\min}(0) = 0$

# B: Building Beaver

Problem author: Jeroen Op de Beek



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:
- $S_{\min}(h) = S_{\min}(h-2) + S_{\min}(h-1) + 1$ ,  $S_{\min}(0) = 0$
- $S_{\max}(h) = 2 \cdot S_{\max}(h-1) + 1$ ,  $S_{\max}(0) = 0$

# B: Building Beaver

Problem author: Jeroen Op de Beek



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:
- $S_{\min}(h) = S_{\min}(h-2) + S_{\min}(h-1) + 1$ ,  $S_{\min}(0) = 0$
- $S_{\max}(h) = 2 \cdot S_{\max}(h-1) + 1$ ,  $S_{\max}(0) = 0$
- Only need  $O(\log(n))$  first values, as both grow exponentially.

## B: Building Beaver

Problem author: Jeroen Op de Beek



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:
- $S_{\min}(h) = S_{\min}(h-2) + S_{\min}(h-1) + 1$ ,  $S_{\min}(0) = 0$
- $S_{\max}(h) = 2 \cdot S_{\max}(h-1) + 1$ ,  $S_{\max}(0) = 0$
- Only need  $O(\log(n))$  first values, as both grow exponentially.
- **Solution (cont.):** To calculate  $\text{avltree}(n, h_{\text{LO}}, h_{\text{HI}})$ , loop over left and right subtree heights  $h_L, h_R$ . Make sure  $|h_L - h_R| \leq 1$  (AVL property), and  $h_{\text{LO}} \leq 1 + \max(h_L, h_R) \leq h_{\text{HI}}$

## B: Building Beaver

Problem author: Jeroen Op de Beek



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:
- $S_{\min}(h) = S_{\min}(h-2) + S_{\min}(h-1) + 1$ ,  $S_{\min}(0) = 0$
- $S_{\max}(h) = 2 \cdot S_{\max}(h-1) + 1$ ,  $S_{\max}(0) = 0$
- Only need  $O(\log(n))$  first values, as both grow exponentially.
- **Solution (cont.):** To calculate  $\text{avltree}(n, h_{\text{LO}}, h_{\text{HI}})$ , loop over left and right subtree heights  $h_L, h_R$ . Make sure  $|h_L - h_R| \leq 1$  (AVL property), and  $h_{\text{LO}} \leq 1 + \max(h_L, h_R) \leq h_{\text{HI}}$
- By using  $S_{\min}(h)$  and  $S_{\max}(h)$ , figure out which  $h_L, h_R$  pairs give the minimum left subtree size  $S_L$ .



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:
- $S_{\min}(h) = S_{\min}(h-2) + S_{\min}(h-1) + 1$ ,  $S_{\min}(0) = 0$
- $S_{\max}(h) = 2 \cdot S_{\max}(h-1) + 1$ ,  $S_{\max}(0) = 0$
- Only need  $O(\log(n))$  first values, as both grow exponentially.
- **Solution (cont.):** To calculate  $\text{avltree}(n, h_{\text{LO}}, h_{\text{HI}})$ , loop over left and right subtree heights  $h_L, h_R$ . Make sure  $|h_L - h_R| \leq 1$  (AVL property), and  $h_{\text{LO}} \leq 1 + \max(h_L, h_R) \leq h_{\text{HI}}$
- By using  $S_{\min}(h)$  and  $S_{\max}(h)$ , figure out which  $h_L, h_R$  pairs give the minimum left subtree size  $S_L$ .
- **Observation:** All  $h_L$ 's reaching the minimum left subtree size form an interval  $[a, b]$ .



## B: Building Beaver

Problem author: Jeroen Op de Beek



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:
- $S_{\min}(h) = S_{\min}(h-2) + S_{\min}(h-1) + 1$ ,  $S_{\min}(0) = 0$
- $S_{\max}(h) = 2 \cdot S_{\max}(h-1) + 1$ ,  $S_{\max}(0) = 0$
- Only need  $O(\log(n))$  first values, as both grow exponentially.
- **Solution (cont.):** To calculate  $\text{avltree}(n, h_{\text{LO}}, h_{\text{HI}})$ , loop over left and right subtree heights  $h_L, h_R$ . Make sure  $|h_L - h_R| \leq 1$  (AVL property), and  $h_{\text{LO}} \leq 1 + \max(h_L, h_R) \leq h_{\text{HI}}$
- By using  $S_{\min}(h)$  and  $S_{\max}(h)$ , figure out which  $h_L, h_R$  pairs give the minimum left subtree size  $S_L$ .
- **Observation:** All  $h_L$ 's reaching the minimum left subtree size form an interval  $[a, b]$ .
- Use  $\text{avltree}(S_L, a, b)$  to find optimal left subtree ordering.

## B: Building Beaver

Problem author: Jeroen Op de Beek



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:
- $S_{\min}(h) = S_{\min}(h-2) + S_{\min}(h-1) + 1$ ,  $S_{\min}(0) = 0$
- $S_{\max}(h) = 2 \cdot S_{\max}(h-1) + 1$ ,  $S_{\max}(0) = 0$
- Only need  $O(\log(n))$  first values, as both grow exponentially.
- **Solution (cont.):** To calculate  $\text{avltree}(n, h_{\text{LO}}, h_{\text{HI}})$ , loop over left and right subtree heights  $h_L, h_R$ . Make sure  $|h_L - h_R| \leq 1$  (AVL property), and  $h_{\text{LO}} \leq 1 + \max(h_L, h_R) \leq h_{\text{HI}}$
- By using  $S_{\min}(h)$  and  $S_{\max}(h)$ , figure out which  $h_L, h_R$  pairs give the minimum left subtree size  $S_L$ .
- **Observation:** All  $h_L$ 's reaching the minimum left subtree size form an interval  $[a, b]$ .
- Use  $\text{avltree}(S_L, a, b)$  to find optimal left subtree ordering.
- Left subtree height  $h_L$  is known, so  $h_R$  subtree heights will again form interval, call  $\text{avltree}(n-1-S_L, u, v)$ , with appropriate  $u, v$ , and reindex with offset.

## B: Building Beaver

Problem author: Jeroen Op de Beek



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:
- $S_{\min}(h) = S_{\min}(h-2) + S_{\min}(h-1) + 1$ ,  $S_{\min}(0) = 0$
- $S_{\max}(h) = 2 \cdot S_{\max}(h-1) + 1$ ,  $S_{\max}(0) = 0$
- Only need  $O(\log(n))$  first values, as both grow exponentially.
- **Solution (cont.):** To calculate  $\text{avltree}(n, h_{\text{LO}}, h_{\text{HI}})$ , loop over left and right subtree heights  $h_L, h_R$ . Make sure  $|h_L - h_R| \leq 1$  (AVL property), and  $h_{\text{LO}} \leq 1 + \max(h_L, h_R) \leq h_{\text{HI}}$
- By using  $S_{\min}(h)$  and  $S_{\max}(h)$ , figure out which  $h_L, h_R$  pairs give the minimum left subtree size  $S_L$ .
- **Observation:** All  $h_L$ 's reaching the minimum left subtree size form an interval  $[a, b]$ .
- Use  $\text{avltree}(S_L, a, b)$  to find optimal left subtree ordering.
- Left subtree height  $h_L$  is known, so  $h_R$  subtree heights will again form interval, call  $\text{avltree}(n-1-S_L, u, v)$ , with appropriate  $u, v$ , and reindex with offset.
- Time complexity:  $T(n) = O(\log(n)) + T(S_L) + T(n-1-S_L) = O(n \log(n))$ .

## B: Building Beaver

Problem author: Jeroen Op de Beek



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:
- $S_{\min}(h) = S_{\min}(h-2) + S_{\min}(h-1) + 1$ ,  $S_{\min}(0) = 0$
- $S_{\max}(h) = 2 \cdot S_{\max}(h-1) + 1$ ,  $S_{\max}(0) = 0$
- Only need  $O(\log(n))$  first values, as both grow exponentially.
- **Solution (cont.):** To calculate  $\text{avltree}(n, h_{\text{LO}}, h_{\text{HI}})$ , loop over left and right subtree heights  $h_L, h_R$ . Make sure  $|h_L - h_R| \leq 1$  (AVL property), and  $h_{\text{LO}} \leq 1 + \max(h_L, h_R) \leq h_{\text{HI}}$
- By using  $S_{\min}(h)$  and  $S_{\max}(h)$ , figure out which  $h_L, h_R$  pairs give the minimum left subtree size  $S_L$ .
- **Observation:** All  $h_L$ 's reaching the minimum left subtree size form an interval  $[a, b]$ .
- Use  $\text{avltree}(S_L, a, b)$  to find optimal left subtree ordering.
- Left subtree height  $h_L$  is known, so  $h_R$  subtree heights will again form interval, call  $\text{avltree}(n-1-S_L, u, v)$ , with appropriate  $u, v$ , and reindex with offset.
- Time complexity:  $T(n) = O(\log(n)) + T(S_L) + T(n-1-S_L) = O(n \log(n))$ .
- **Bonus:** Can prove because recursion tree is AVL tree,  $T(n) = \Theta(n)$ .

## B: Building Beaver

Problem author: Jeroen Op de Beek



- **Observation:** Can calculate minimum and maximum subtree size of AVL BST of height  $h$  with recursion:
- $S_{\min}(h) = S_{\min}(h-2) + S_{\min}(h-1) + 1$ ,  $S_{\min}(0) = 0$
- $S_{\max}(h) = 2 \cdot S_{\max}(h-1) + 1$ ,  $S_{\max}(0) = 0$
- Only need  $O(\log(n))$  first values, as both grow exponentially.
- **Solution (cont.):** To calculate  $\text{avltree}(n, h_{\text{LO}}, h_{\text{HI}})$ , loop over left and right subtree heights  $h_L, h_R$ . Make sure  $|h_L - h_R| \leq 1$  (AVL property), and  $h_{\text{LO}} \leq 1 + \max(h_L, h_R) \leq h_{\text{HI}}$
- By using  $S_{\min}(h)$  and  $S_{\max}(h)$ , figure out which  $h_L, h_R$  pairs give the minimum left subtree size  $S_L$ .
- **Observation:** All  $h_L$ 's reaching the minimum left subtree size form an interval  $[a, b]$ .
- Use  $\text{avltree}(S_L, a, b)$  to find optimal left subtree ordering.
- Left subtree height  $h_L$  is known, so  $h_R$  subtree heights will again form interval, call  $\text{avltree}(n-1-S_L, u, v)$ , with appropriate  $u, v$ , and reindex with offset.
- Time complexity:  $T(n) = O(\log(n)) + T(S_L) + T(n-1-S_L) = O(n \log(n))$ .
- **Bonus:** Can prove because recursion tree is AVL tree,  $T(n) = \Theta(n)$ .

Statistics: 54 submissions, 0 accepted, 40 unknown