获得的答案 **返回**

A binary search tree is built in such a way that the right child must be greater than the parent and the left child must be less than the parent. Therefore, sort a set of numbers by building a binary search tree and print using tree walk (or tree traversing).

The algorithm to sort and print set of numbers using binary search tree is as follows:

TREE-SORT (*A*)

1. **for** *i* =1 **to** *n*

2. TREE-INSERT(*T*, *A*[*i*]) //let *T* be an empty binary search free

3. INORDER-TREE-WALK(*T.root*)

TREE-INSERT (*T, z*)

1. *y* = NIL

2. *x* = *T.root*

3. **while** *x* ≠ NIL

4. *y* = *x*

5. **if** *z.key* < *x.key*

6. *x* = *x.left*

7. **else** *x* = *x.right*

8. *z.p* = *y*

9. **if** *y* = = NIL

10. *T.root* = *z* // tree T was empty

11. **elseif** *z.key* < *y.key*

12. *y.left* = *z*

13. **else** *y.right* = *z*

INORDER-TREE-WALK (*x*)

1. **if** *x* ≠ NIL

2. INORDER-TREE-WALK (*x.left*)

3. print *x.key*

4. INORDER-TREE-WALK (*x.right*)

Worst case running time analysis for TREE-SORT:

If the linear chain of numbers (numbers in ascending or descending order) is to be inserted, TREE-INSERT takes worst case running time.

That is, in the worst case, TREE-INSERT checks the nodes proportional to height of the tree and inserts a node in $\Theta(n)$ time.

If array *A* contains 'n' numbers in ascending order or descending order in each iteration of for loop, TREE-INSERT executes in time proportional to height of the tree.

Thus, the for loop executes overall in (1(1st iteration)+2(2nd iteration)+...+n(n th iteration))

$$1+2+\ldots+n=\frac{n(n+1)}{2}=\frac{n^2+n}{2}=\Theta(n^2)$$

Hence, the worst case running time for TREE-SORT is $\Theta(n^2)$.

Best case running time for TREE-SORT:

If the set of nonlinear numbers (numbers neither in ascending nor in descending order) is to be inserted, TREE-INSERT takes best case running time.

That is TREE-INSERT inserts a node in $\Theta(\log n)$ time.

If array *A* contains 'n' nonlinear numbers, the for loop executes n times and each time TREE-INSERT executes in log n time. Overall, the for loop executes $n \times \log n$ times.

Hence, the best case running time for TREE-SORT is $\Theta(n \log n)$.