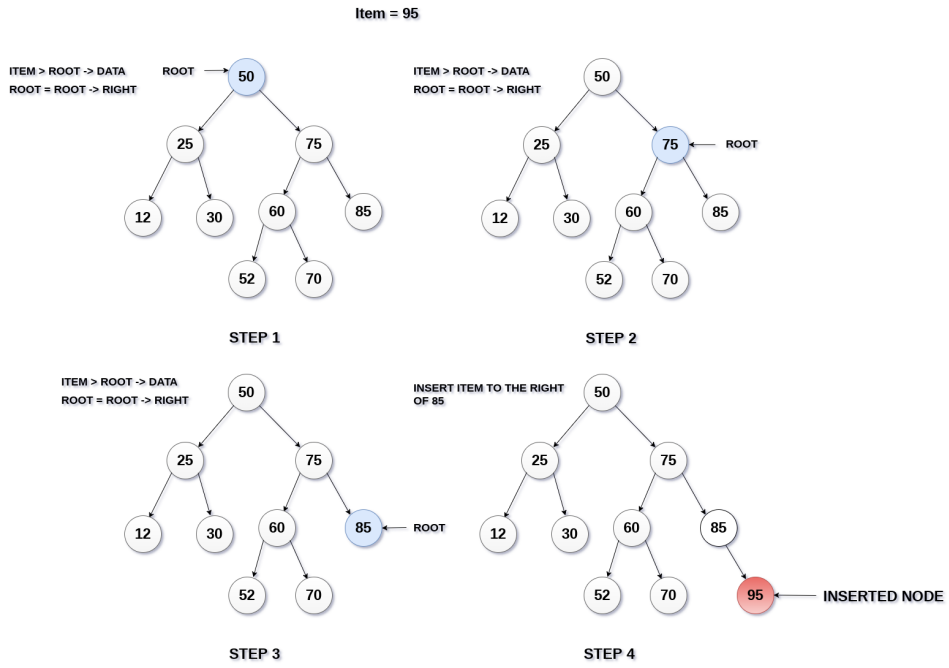
Insertion

Insert function is used to add a new element in a binary search tree at appropriate location. Insert function is to be designed in such a way that, it must node violate the property of binary search tree at each value.

1. Allocate the memory for tree.
2. Set the data part to the value and set the left and right pointer of tree, point to NULL.
3. If the item to be inserted, will be the first element of the tree, then the left and right of this node will point to NULL.
4. Else, check if the item is less than the root element of the tree, if this is true, then recursively perform this operation with the left of the root.
5. If this is false, then perform this operation recursively with the right sub-tree of the root.

Insert (TREE, ITEM)

- **Step 1:** IF TREE = NULL
  Allocate memory for TREE
  SET TREE -> DATA = ITEM
  SET TREE -> LEFT = TREE -> RIGHT = NULL
  ELSE
  IF ITEM < TREE -> DATA
  Insert(TREE -> LEFT, ITEM)
  ELSE
  Insert(TREE -> RIGHT, ITEM)
  [END OF IF]
  [END OF IF]
- **Step 2:** END



C Function

```
#include<stdio.h>
#include<stdlib.h>
void insert(int);
struct node
{
   int data;
   struct node *left;
   struct node *right;
};
struct node *root;
void main ()
{
   int choice,item;
   do
   {
      printf("\nEnter the item which you want to insert?\n");
      scanf("%d",&item);
      insert(item);
      printf("\nPress 0 to insert more ?\n");
      scanf("%d",&choice);
   }while(choice == 0);
}
void insert(int item)
{
   struct node *ptr, *parentptr , *nodeptr;
   ptr = (struct node *) malloc(sizeof (struct node));
   if(ptr == NULL)
   {
      printf("can't insert");
```

```
    }
    else
    {
    ptr -> data = item;
    ptr -> left = NULL;
    ptr -> right = NULL;
    if(root == NULL)
    {
       root = ptr;
       root -> left = NULL;
       root -> right = NULL;
    }
    else
    {
       parentptr = NULL;
       nodeptr = root;
       while(nodeptr != NULL)
       {
          parentptr = nodeptr;
          if(item < nodeptr->data)
          {
             nodeptr = nodeptr -> left;
          }
          else
          {
             nodeptr = nodeptr -> right;
          }
       }
       if(item < parentptr -> data)
       {
          parentptr -> left = ptr;
       }
       else
       {
          parentptr -> right = ptr;
       }
    }
    printf("Node Inserted");
    }
}
```

Output

```
Enter the item which you want to insert?
12
Node Inserted
Press 0 to insert more ?
0

Enter the item which you want to insert?
23
Node Inserted
```

```
Press 0 to insert more ?
1
```

← prev next →

Please Share

Learn Latest Tutorials

C. Graphics | Automata | Testing | NumPy

Verbal A. | AWS

Preparation

Aptitude | Reasoning | Verbal A. | Interview

B.Tech / MCA

DBMS | DS | DAA | OS


C. Network


Compiler D.


COA

D. Math.

Web Tech.


Cyber Sec.


C


C++


Java

.Net


Python

Programs


Control S.