**Data Structures**

Implementation of Binary Search Tree

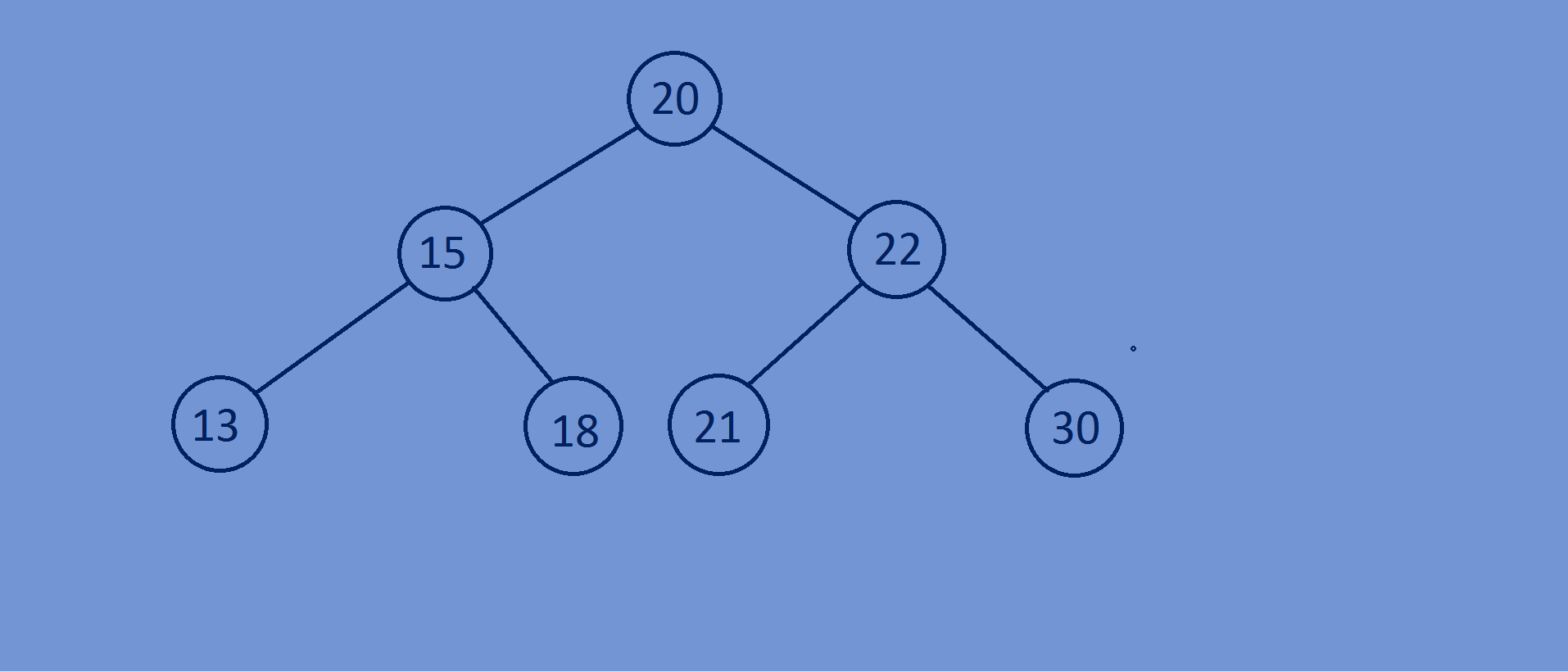
**Binary Search Tree:**

* A Binary Search tree is a binary tree in which value of left sub-tree node is

Less than or equal to its parent nodes value and the value of right sub-tree

Node is greater than its parent nodes value,here for each node,value of all

the nodes in left subtree is lesser and value of all the nodes in right subtree

****is greater.

**Implementation of Binary Search Tree:**

* If root is equal to NULL we make the newly created node(newnode) as the root.
* If the data in the newnode is less than or equal to the data of the root node then

We make a recursive call of the function to insert the newnode in the left

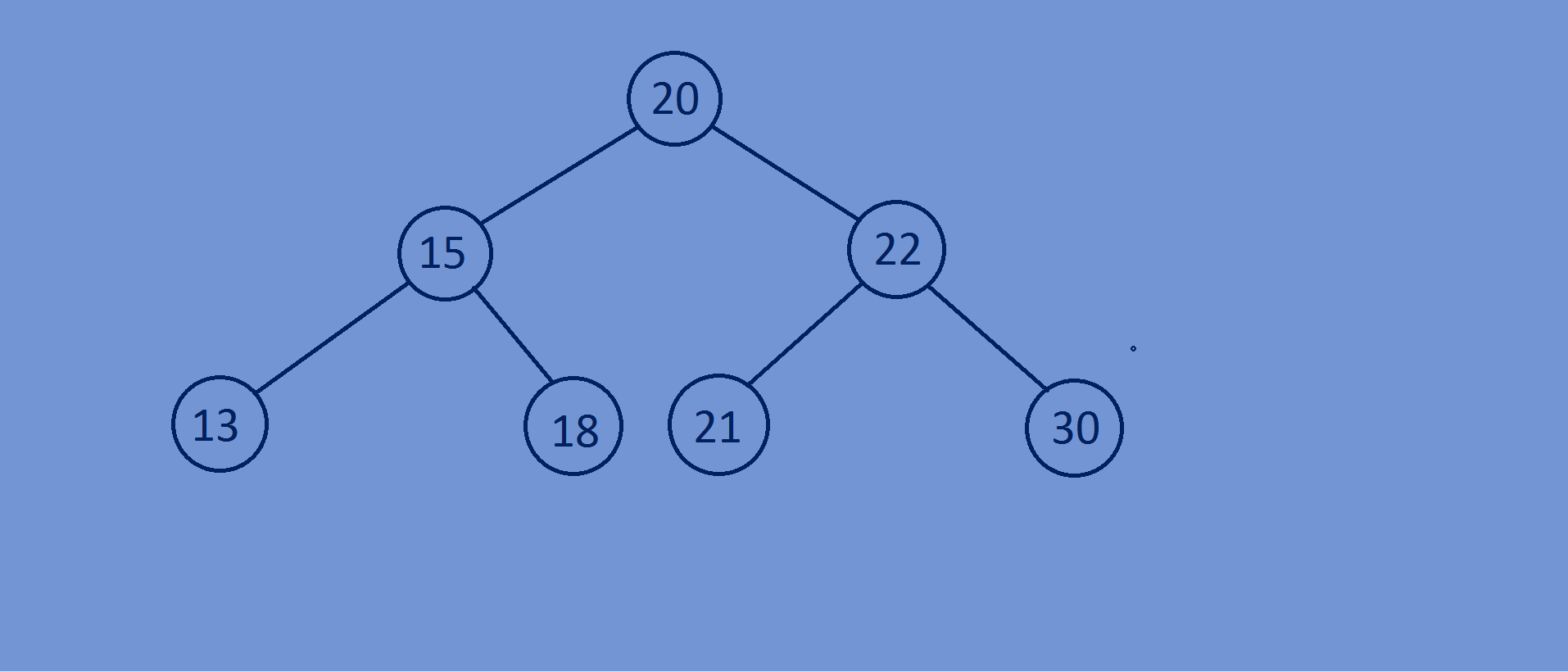
Subtree and then set the new root of the left subtree in root->left.

root->left=InsertNodeintoBStree(root->left,newnode).

* If the data in the newnode is greater than the data of the root node then

We make a recursive call of the function to insert the newnode in the right

Subtree and then set the new root of the right subtree in root->right.

**** root->right=InsertNodeintoBStree(root->right,newnode).

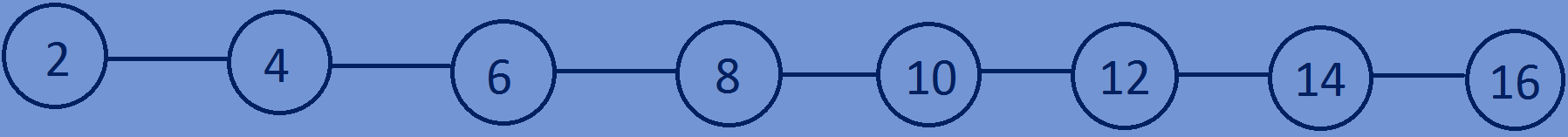
**Function for Creating of a Binary Search Tree:**

BStree \*InsertNodeintoBStree(BStree \*root,BStree \*newnode){  
 **if**(root==NULL){  
 root=newnode;  
 **return** root;  
 }  
 **if**(newnode->data<=root->data){  
 root->left=InsertNodeintoBStree(root->left,newnode);  
 }  
 **else if**(newnode->data>root->data){  
 root->right=InsertNodeintoBStree(root->right,newnode);  
 }  
 **return** root;  
}  
BStree \*implementBStree(BStree \*root,**int** n){  
 **for**(**int** i=1;i<=n;i++){  
 BStree \*newnode=(BStree\*)malloc(**sizeof**(BStree));  
 newnode->data=i\*2;  
 newnode->left=NULL;  
 newnode->right=NULL;  
 root=InsertNodeintoBStree(root,newnode);  
 }  
 **return** root;  
}

**Whole Program:**

#include<stdio.h>  
#include<stdlib.h>  
//creating a node  
**typedef struct** BStree{  
 **int** data;  
 **struct** BStree \*left;  
 **struct** BStree \*right;  
}BStree;  
//Function for inserting nodes into a binary search tree.  
BStree \*InsertNodeintoBStree(BStree \*root,BStree \*newnode){  
 **if**(root==NULL){  
 root=newnode;  
 **return** root;  
 }  
 **if**(newnode->data<=root->data){  
 root->left=InsertNodeintoBStree(root->left,newnode);  
 }  
 **else if**(newnode->data>root->data){  
 root->right=InsertNodeintoBStree(root->right,newnode);  
 }  
 **return** root;  
}  
//creating new nodes  
BStree \*implementBStree(BStree \*root,**int** n){  
 **for**(**int** i=1;i<=n;i++){  
 BStree \*newnode=(BStree\*)malloc(**sizeof**(BStree));  
 newnode->data=i\*2;  
 newnode->left=NULL;  
 newnode->right=NULL;  
 root=InsertNodeintoBStree(root,newnode);  
 }  
 **return** root;  
}  
//printing the data in ascending order(inorder traversal)  
**void** printAscendingorder(BStree \*root){  
 **if**(root){  
 printAscendingorder(root->left);  
 printf("%d ",root->data);  
 printAscendingorder(root->right);  
 }  
}  
//main function  
**int** main(){  
 BStree \*root=NULL;  
 root=implementBStree(root,8);  
 printAscendingorder(root);  
 **return** 0;  
}

**Output:**

2 4 6 8 10 12 14 16