```
/******************************************************************************

                              Online C++ Compiler.
               Code, Compile, Run and Debug C++ program online.
Write your code in this editor and press "Run" button to compile and execute it.

*******************************************************************************/
/*
Subject: DSA Laboratory
Practical No:09
Title: A C++ Program to Implement Dictionary using Height-Balanced Tree.
    Input: Keywords and Meanings
    Output:
          a) Store Keywords and Meanings in Height-Balanced Tree.
          b) Display Keywords and Meanings in Height-Balanced Tree.
          c) Display a Sorted List of Keywords and Meanings.
          d) Display Number of Comparisons required to find a keyword.
*/
        //.......Header Files
#include <iostream>
using namespace std;

            //.......Structure for HBTree Node
struct HBTNode
{
   int Key;
   char Mean[10];

   HBTNode *left;
   HBTNode *right;
}*Root;

              //.....Function to Store Keywords and Meanings in Height-Balanced Tree.
void create_HBT()
{
   int i;
   int nodes;
   int done;
   struct HBTNode *Newnode, *current;

   cout<<"\n\n Enter the no of nodes to insert in HBT.? : ";
   cin>>nodes;

   for(i=0; i<nodes; i++)
   {
      Newnode = new struct HBTNode;   //...Memory Allocation

      cout<<"\n\t Enter Keyword: ";   //...store Keys
      cin>>Newnode->Key;
      cout<<"\n\t Enter Meaning: ";   //...store Meanings
      cin>>Newnode->Mean;
      Newnode->left = NULL;           //...Left and Right pointers initially NULL
    Newnode->right = NULL;

    if(Root == NULL)
    {
     Root = Newnode;
    }
    else
    {
      done = 0;
      current = Root;
      while(!done)
      {
        if(Newnode->Key < current->Key)
        {
         if(current->left == NULL)
         {
           current->left = Newnode;
           done = 1;
         }
         else
           current = current->left;
        }
        else
        {
         if(current->right == NULL)
         {
           current->right = Newnode;
           done = 1;
         }
         else
           current = current->right;
        }
      }//end of while

    }//end of else

  }//end of for

}//end of function

         //.....Function to Display Keywords and Meanings in Height-Balanced Tree.
void display_HBT(struct HBTNode *root)
{
  if(root)                  //....Pre-order Display
  {
    cout<<"\n\t"<<root->Key<<" - "<<root->Mean; //...Data
    display_HBT(root->left);             //...Left
    display_HBT(root->right);            //...Right
  }
}
```

```
    //.....Function to display a Sorted List of Keywords and Meanings.
void Sorted_List(struct HBTNode *root)
{
   if(root)                          //....In-order Display
   {
     Sorted_List(root->left);                     //...Left
     cout<<"\n\t"<<root->Key<<" - "<<root->Mean; //...Data
     Sorted_List(root->right);                    //...Right
   }
}

          //.....Function to Display Number of Comparisons required to find a keyword.";
void Find_Keyword(int key)
{
  int comp = 0;
  int level = 0;
  int done;
  struct HBTNode *current;

  done = 0;
  current = Root;
  while(!done)
  {
    if(key < current->Key)
    {
       current = current->left;
       level++;
       comp++;
    }
    else if(key > current->Key)
    {
       current = current->right;
       level++;
       comp++;
    }
    else
    {
      done = 1;
      comp++;
      cout<<"\n\t Key : "<<key;
      cout<<"\n\t Found at Level: "<<level;
      cout<<"\n\t No. of Comparisons: "<<comp;
    }

   }//end of while


}
```

```
        //.......Main Function
int main()
{
    cout<<"\n -------***A C++ Program to Implement Dictionary using Height-Balanced
Tree.***-------\n";

    cout<<"\n 1. Store Keywords and Meanings in Height-Balanced Tree.";
    Root = NULL;
    create_HBT();

    cout<<"\n 2. Display Keywords and Meanings in Height-Balanced Tree.";
    cout<<"\n Keyword - Meaning";
    display_HBT(Root);

    cout<<"\n 3. Display a Sorted List of Keywords and Meanings.";
    cout<<"\n Keyword - Meaning";
    Sorted_List(Root);

    cout<<"\n 4. Display Number of Comparisons required to find a keyword.";
    Find_Keyword(1);

    return 0;
}
```

```
/*-------------------------------OUTPUT------------------------------------

-------***A C++ Program to Implement Dictionary using Height-Balanced Tree.***-------

 1. Store Keywords and Meanings in Height-Balanced Tree.

 Enter the no of nodes to insert in HBT.? : 5

        Enter Keyword: 3

        Enter Meaning: Three

        Enter Keyword: 2

        Enter Meaning: Two

        Enter Keyword: 4

        Enter Meaning: Four

        Enter Keyword: 1

        Enter Meaning: One

        Enter Keyword: 5

        Enter Meaning: Five
 2. Display Keywords and Meanings in Height-Balanced Tree.
 Keyword - Meaning
        3 - Three
        2 - Two
        1 - One
        4 - Four
        5 - Five
 3. Display a Sorted List of Keywords and Meanings.
 Keyword - Meaning
        1 - One
        2 - Two
        3 - Three
        4 - Four
        5 - Five
 4. Display Number of Comparisons required to find a keyword.
         Key : 1
         Found at Level: 2
         No. of Comparisons: 3

...Program finished with exit code 0
Press ENTER to exit console.
*/
```