/*NAME – ABHISHEK

SID – 21102044

*/

```cpp
#include<iostream>
using namespace std;
struct node
{
    int value;
    node * pLeft;
    node * pRight;
    node(int val = 0)
    {
        value = val;
        pRight = NULL;
        pLeft = NULL;
    }
};

void insert(node ** root, int val)
{
    if(*root == NULL)
        *root = new node(val);
    else if((*root)->value <= val)
        insert(&((*root)->pRight), val);
    else if((*root)->value > val)
        insert(&((*root)->pLeft), val);
}

node * getBST(int * arr, int size)
{
    node * root = NULL;
    for(int i = 0; i < size; i++)
        insert(&root, arr[i]);
    return root;
}

void inOrderTraversal(node * root)
{
    if(root && root->pLeft)
```

```
        inOrderTraversal(root->pLeft);
    if(root)
        cout<<root->value<<" ";
    if(root && root->pRight)
        inOrderTraversal(root->pRight);

}

int deletion_from_Array(int arr[], int n, int x)
{
    int i;
    for (i=0; i<n; i++)
        if (arr[i] == x)    break;


    if (i < n){
        n = n - 1;
        for (int j=i; j<n; j++)
            arr[j] = arr[j+1];
    }

    return n;
}

int successor(node * root) {
  root = root -> pRight;
  while (root -> pLeft != NULL) root = root -> pLeft;
  return root -> value;
}

int predecessor(node * root) {
  root = root -> pLeft;
  while (root -> pRight != NULL) root = root -> pRight;
  return root -> value;
}

node* deleteNodeFromBST(node * root, int key)
{
  if (root == NULL) return NULL;
  if (key > root -> value) root -> pRight = deleteNodeFromBST(root -> pRight,
key);
  else if (key < root -> value) root -> pLeft = deleteNodeFromBST(root ->
pLeft, key);
  else {
    if (root -> pLeft == NULL && root -> pRight == NULL) root = NULL;
    else if (root -> pRight != NULL) {
      root -> value = successor(root);
      root -> pRight = deleteNodeFromBST(root -> pRight, root -> value);
    } else {
      root -> value = predecessor(root);
      root -> pLeft = deleteNodeFromBST(root -> pLeft, root -> value);
    }
  }
  return root;
}

void printarray(int arr[],int n){
    for(int i = 0; i< n ; i++) cout<<arr[i]<<" ";

}

int main()
{
    int arr[] = {10,5,15,5,6,7,8,89};
    node * root = getBST(arr, sizeof(arr)/sizeof(int));
    inOrderTraversal(root);
    cout<<endl;
    deleteNodeFromBST(root,6);
    inOrderTraversal(root);
    cout<<endl;
    deletion_from_Array(arr,sizeof(arr)/sizeof(int),6);
    printarray(arr,sizeof(arr)/sizeof(int)-1);



    return 0;
}
```

/* Output –

5 5 6 7 8 10 15 89

5 5 7 8 10 15 89

10 5 15 5 7 8 89

*/

/*

SPACE COMPLEXITIES :

Both the array and the Binary search tree in the above case have the space complexity O(n)

*/