

Assignment 08

Name: Deepak Singh

Roll No. 20bcs102

Solution 1

Code:

First Element as pivot

```
#include <iostream>
using namespace std;
int partition(int arr[], int low, int high) {
    int pivot = arr[low];
    int cnt = 0;
    for (int i = low + 1; i <= high; i++) {
        if (arr[i] <= pivot) {
            cnt++;
        }
    }
    int pivotindex = low + cnt;
    swap(arr[pivotindex], arr[low]);
    int i = low, j = high;
    while (i < pivotindex && j > pivotindex) {
        while (arr[i] <= pivot) {
            i++;
        }
        while (arr[j] >= pivot) {
            j--;
        }
        if (i < pivotindex && j > pivotindex) {
            swap(arr[i++], arr[j--]);
        }
    }
    return pivotindex;
}
```

```

void quick_sort(int arr[], int low, int high) {
    if (low < high) {
        int p1 = partition(arr, low, high);
        quick_sort(arr, low, p1 - 1);
        quick_sort(arr, p1 + 1, high);
    }
}

int main() {
    int n;
    cin >> n;
    int arr[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
    quick_sort(arr, 0, n - 1);
    cout<<"Sorted Array: "<<endl;
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    return 0;
}

```

Output:

```

deepak@deepak in ~/Desktop/pgm/p1 took 6ms
λ cd "/home/deepak/Desktop/pgm/p1/" && g++ 81.cpp -o 81 && "/home/deepak/Desktop/pgm/p1/"81
10
45 854 -78 5894 20 1 20 854 0 0
Sorted Array:
-78 0 0 1 20 20 45 5894 854 854

```

Code:

Middle Element as pivot

```
#include <iostream>
```

```
using namespace std;
```

```

void quick_sort(int arr[], int low, int high) {

    int i = low, j = high;

    int pivot = arr[(low + high) / 2];

    while (i <= j) {

        while (arr[i] < pivot) {

            i++;

        }

        while (arr[j] > pivot) {

            j--;

        }

        if (i <= j) {

            swap(arr[i], arr[j]);

            i++;

            j--;

        }

    }

    if (low < j) {

        quick_sort(arr, low, j);

    }

    if (i < high) {

        quick_sort(arr, i, high);

    }
}

```

```

}

int main() {

    int n;

    cin >> n;

    int arr[n];

    for (int i = 0; i < n; i++) {

        cin >> arr[i];

    }

    quick_sort(arr, 0, n - 1);

    cout<<"Sorted Array:"<<endl;

    for (int i = 0; i < n; i++) {

        cout << arr[i] << " ";

    }

    return 0;

}

```

Output:

```

deepak@deepak in ~/Desktop/pgm/p1 took 7ms
λ cd "/home/deepak/Desktop/pgm/p1/" && g++ tempCodeRunnerFile.cpp -o te
10
45 854 -78 5894 20 1 20 854 0 1
Sorted Array:
-78 0 1 1 20 20 45 854 854 5894

```

Code:

Random Element as pivot

```
#include <iostream>

using namespace std;

void quick_sort(int arr[], int low, int high) {
    int i = low, j = high;
    int pivot = arr[(low + high) / 2];
    while (i <= j) {
        while (arr[i] < pivot) {
            i++;
        }
        while (arr[j] > pivot) {
            j--;
        }
        if (i <= j) {
            swap(arr[i], arr[j]);
            i++;
            j--;
        }
    }
    if (low < j) {
        quick_sort(arr, low, j);
    }
    if (i < high) {
        quick_sort(arr, i, high);
    }
}

int main() {
    int n;
    cin >> n;
    int arr[n];
    for (int i = 0; i < n; i++) {
```

```

        cin >> arr[i];
    }
    quick_sort(arr, 0, n - 1);
    cout << "Sorted Array:" << endl;
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    return 0;
}

```

Output:

```

λ cd "/home/deepak/Desktop/pgm/p1/" && g++ 813.cpp -o 813 && "/home/deepak/Desktop/pgm/p1/"813
10
45 854 -78 5894 20 1 20 5894 63 45
Sorted Array:
-78 1 20 20 45 45 63 854 5894 5894

```

Solution 2

Code:

```

#include <iostream>

using namespace std;

class treeNode {
public:

    int value;

    treeNode *left;

    treeNode *right;

    treeNode(int _value) {

        value = _value;

        left = NULL;

        right = NULL;
    }
}

```

```

    }
};

void addNode(treeNode *root, int _value) {

    if (root == NULL) {

        root = new treeNode(_value);

    } else if (_value < root->value) {

        if (root->left == NULL) {

            root->left = new treeNode(_value);

        } else {

            addNode(root->left, _value);

        }

    } else {

        if (root->right == NULL) {

            root->right = new treeNode(_value);

        } else {

            addNode(root->right, _value);

        }

    }

}

void display(treeNode *root) {

    if (root == NULL) {

        return;

    }

    display(root->left);

```

```

    cout << root->value << "\\t";
    display(root->right);
}

void preOrderTraversal(treeNode *root) {
    if (root != NULL) {
        cout << root->value << "\\t";
        preOrderTraversal(root->left);
        preOrderTraversal(root->right);
    }
}

void postOrderTraversal(treeNode *root) {
    if (root != NULL) {
        preOrderTraversal(root->left);
        preOrderTraversal(root->right);
        cout << root->value << "\\t";
    }
}

int main() {
    int arrOfElements[9] = {15, 85, 22, 562, 415, 12, 30, 96, 856};
    treeNode *root = NULL;
    for (int num : arrOfElements) {
        addNode(root, num);
    }

    cout << "Preorder Traversal is: ";

```



```

preOrderTraversal(root);

cout << "\nPostorder Traversal is: ";

postOrderTraversal(root);

cout << "\nInorder Traversal is: ";

display(root);

cout << endl;

return 0;
}

```

Output:

```

deepak@deepak in ~/Desktop/pgm/p1 took 7ms
λ cd "/home/deepak/Desktop/pgm/p1/" && g++ 822.cpp -o 822 && "/home/deepak/Desktop/pgm/p1/"822
Preorder Traversal is: 15      12      85      22      30      562      415      96      856
Postorder Traversal is: 12      85      22      30      562      415      96      856      15
Inorder Traversal is: 12      15      22      30      85      96      415      562      856

```

Solution 3

Code:

```

#include <iostream>

using namespace std;

struct Node {

    int data;

    struct Node *left;

    struct Node *right;

    Node(int val) {

        data = val;

        left = NULL;

        right = NULL;
    }
}

```

```

    }
};

int countNodes(Node *root) {
    if (root == NULL) {
        return 0;
    }

    return 1 + countNodes(root->left) + countNodes(root->right);
}

int main() {
    Node *root = new Node(10);
    root->left = new Node(12);
    root->right = new Node(53);
    root->left->left = new Node(74);
    root->left->right = new Node(35);
    root->right->left = new Node(76);
    root->right->right = new Node(17);
    root->right->right->right = new Node(78);

    int ans = countNodes(root);

    cout << ans << endl;

    return ans;
}

```

Output:

```

deepak@deepak in ~/Desktop/pgm/p1 took 6ms
λ cd "/home/deepak/Desktop/pgm/p1/" && g++ 83.cpp -o 83 && "/home/deepak/Desktop/pgm/p1/"83
8

```

Solution 4

Code:

```
#include <iostream>

using namespace std;

class TreeNode {
public:
    int val;
    TreeNode *left;
    TreeNode *right;
    TreeNode *parent;
};

void insert(TreeNode *&node, int key) {
    if (node == NULL) {
        node = new TreeNode;
        node->val = key;
        node->left = NULL;
        node->right = NULL;
        node->parent = NULL;
    } else if (node->val < key) {
        insert(node->right, key);
        node->right->parent = node;
    } else {
        insert(node->left, key);
        node->left->parent = node;
    }
}
```

```

    }
}

void display(TreeNode *root) {

    if (root == NULL) {

        return;

    }

    display(root->left);

    cout << root->val << "\t";

    display(root->right);

}

TreeNode *searchBST(TreeNode *root, int val) {

    if (root == NULL)

        return NULL;

    if (root->val == val) {

        return root;

    };

    if (root->val > val)

        return searchBST(root->left, val);

    return searchBST(root->right, val);

}

void checkType(TreeNode *root, int val) {

    TreeNode *node = searchBST(root, val);

    if (node == NULL) {

        cout << "Node is not present in tree" << endl;
    }
}

```

```

} else {

    if (node->parent == NULL) {

        cout << "Node is a root node and tree is :";

        display(root);

    } else if (node->left == NULL and node->right == NULL) {

        cout << "Node is leaf node having value " << val << endl;

    } else {

        cout << "Node is an internal node having";

        if (node->left == NULL and node->right != NULL) {

            cout << "right child is " << node->right->val << endl;

        } else if (node->left != NULL and node->right == NULL) {

            cout << "left child is " << node->left->val << endl;

        } else {

            cout << "left child is " << node->left->val << "and right child is "

                << node->right->val << endl;

        }

    }

}

}

}

int main() {

    TreeNode *root = NULL;

    insert(root, 11);

    insert(root, 2);

    insert(root, 41);

```

```

insert(root, -117);

insert(root, 138);

insert(root, 45);

insert(root, 38);

cout << endl;

checkType(root, 41);

checkType(root, 38);

cout << endl;

checkType(root, 100);

return 0;
}

```

Output:

```

deepak@deepak in ~/Desktop/pgm/p1 took 7ms
λ cd "/home/deepak/Desktop/pgm/p1/" && g++ 84.cpp -o 84 && "/home/deepak/Desktop/pgm/p1/"84

Node is an internal node having left child is 38 and right child is 138
Node is leaf node having value 38

Node is not present in tree

```