**NAME: ALI HASSAN CLASS: BS.IT 3A ENROLL.NO: 03-135211-005**

**TASK 1:**

#include <iostream>

using namespace std;

class node

{

public:

int data;

node\* left;

node\* right;

};

int maxDepth(node\* node)

{

if (node == NULL)

return 0;

else

{

int left\_depth = maxDepth(node->left);

int right\_depth = maxDepth(node->right);

if (left\_depth > right\_depth)

return(left\_depth + 1);

else return(right\_depth + 1);

}

}

node\* newNode(int data)

{

node\* Node = new node();

Node->data = data;

Node->left = NULL;

Node->right = NULL;

return(Node);

}

int main()

{

node\* root = newNode(1);

root->left = newNode(2);

root->right = newNode(3);

root->left->left = newNode(4);

root->left->right = newNode(5);

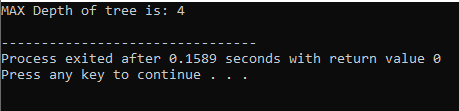
root->left->left->left = newNode(6);

cout << "MAX Depth of tree is: " << maxDepth(root) << endl;

return 0;

}

**Output:**

****

**Task 2:**

#include <iostream>

using namespace std;

class node

{

public:

int data;

node\* left;

node\* right;

};

int minValue(node\* root)

{

if (root == NULL)

{

return INT\_MAX;

}

int result = root->data;

int left = minValue(root->left);

int right = minValue(root->right);

if (left < result)

{

result = left;

}

if (right < result)

{

result = right;

}

return result;

}

node\* newNode(int data)

{

node\* Node = new node();

Node->data = data;

Node->left = NULL;

Node->right = NULL;

return(Node);

}

int main()

{

node\* root = newNode(11);

root->left = newNode(2);

root->right = newNode(3);

root->left->left = newNode(4);

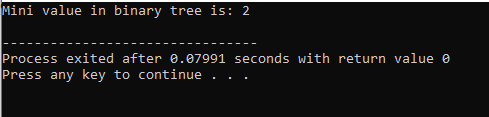
root->left->right = newNode(5);

cout << "Mini value in binary tree is: " << minValue(root) << endl;

return 0;

}

**Output:**

****

**Task 3:**

#include <iostream>

using namespace std;

class node

{

public:

int data;

node\* left;

node\* right;

};

void printPathsRecur(node\* node, int path[], int pathLen);

void printArray(int ints[], int len);

void printPaths(node\* node)

{

int path[1000];

printPathsRecur(node, path, 0);

}

void printPathsRecur(node\* node, int path[], int pathLen)

{

if (node == NULL)

return;

path[pathLen] = node->data;

pathLen++;

if (node->left == NULL && node->right == NULL)

{

printArray(path, pathLen);

}

else

{

printPathsRecur(node->left, path, pathLen);

printPathsRecur(node->right, path, pathLen);

}

}

void printArray(int ints[], int len)

{

int i;

for (i = 0; i < len; i++)

{

cout << ints[i] << " ";

}

cout << endl;

}

node\* newNode(int data)

{

node\* Node = new node();

Node->data = data;

Node->left = NULL;

Node->right = NULL;

return(Node);

}

int main()

{

node\* root = newNode(1);

root->left = newNode(4);

root->right = newNode(5);

root->left->left = newNode(3);

root->left->right = newNode(2);

printPaths(root);

return 0;

}

**Output:**

