

DATE:-21/09/22

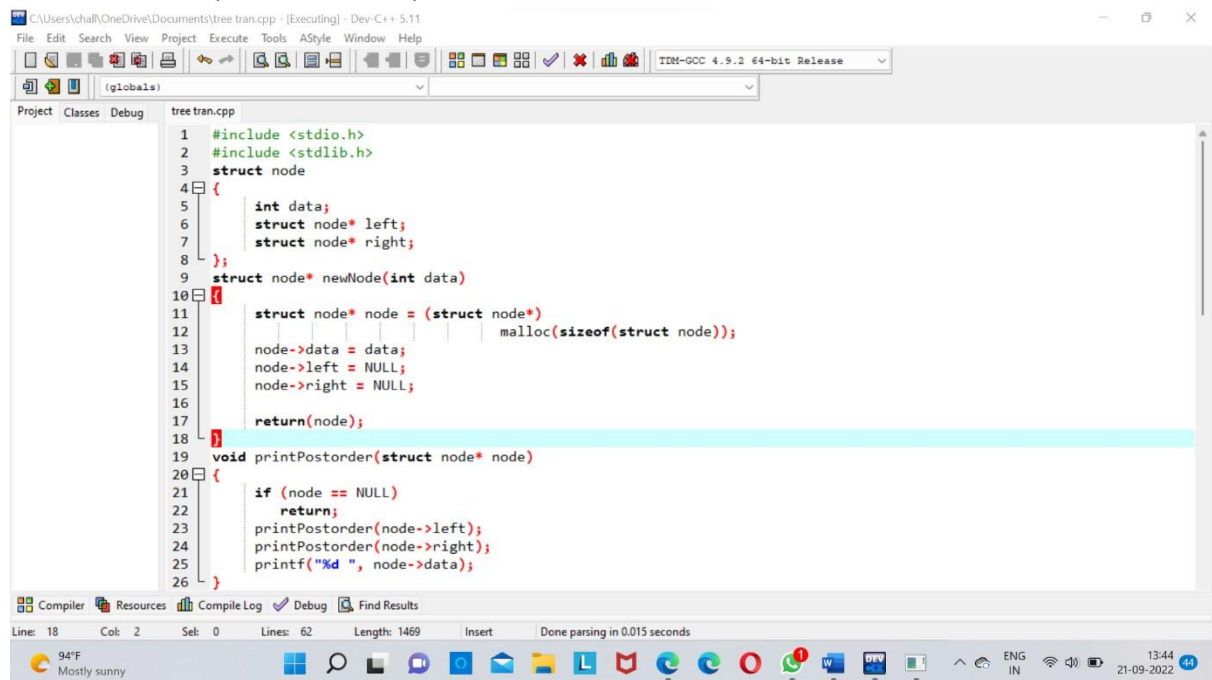
COURSE NAME:-DATA STRUCTURES FOR EXPRESSION EVALUATION

COURSE CODE:-CSA0374

NAME OF THE STUDENT:-CH.INDHU PRIYA

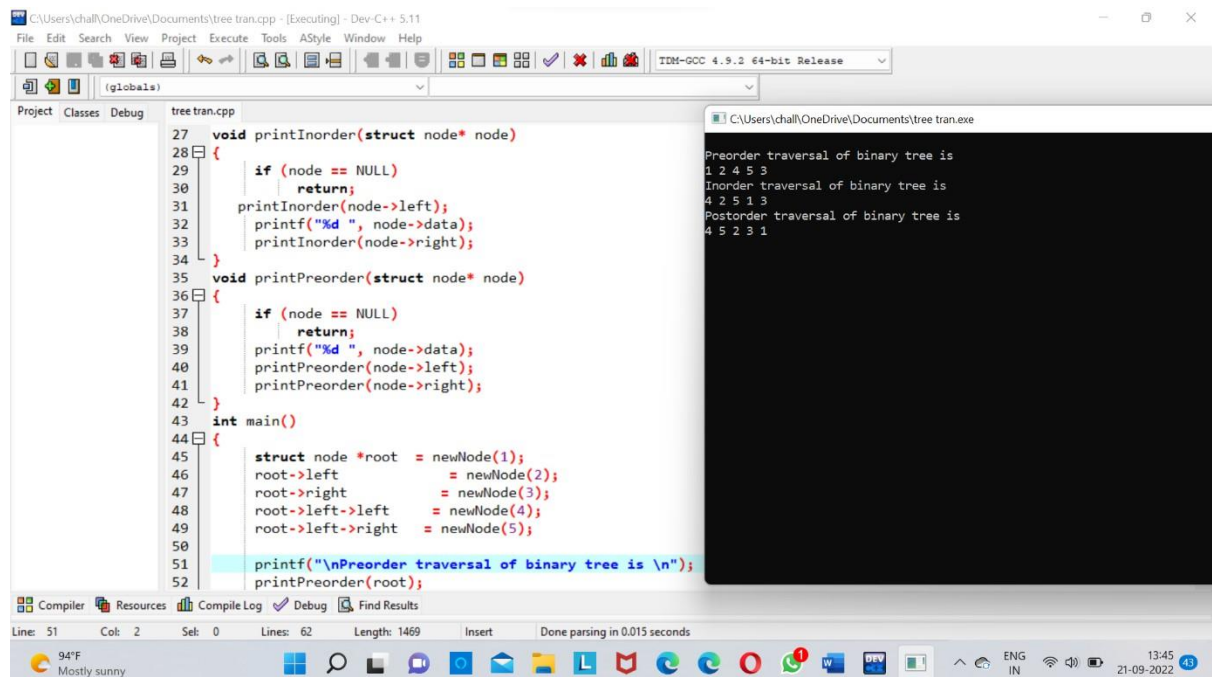
REGNO:-192111191

EXPERIMENT:-15(TREE TRANSVAL)



The screenshot shows the Dev-C++ IDE with the file 'tree tran.cpp' open. The code defines a binary tree node structure and functions for creating a new node and printing the postorder traversal. The node structure has an integer 'data' and pointers to 'left' and 'right' nodes. The 'newNode' function takes an integer and returns a pointer to a new node. The 'printPostorder' function recursively prints the data of the left child, then the right child, and finally the root node.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 struct node
4 {
5     int data;
6     struct node* left;
7     struct node* right;
8 };
9 struct node* newNode(int data)
10 {
11     struct node* node = (struct node*)
12         malloc(sizeof(struct node));
13     node->data = data;
14     node->left = NULL;
15     node->right = NULL;
16     return(node);
17 }
18 void printPostorder(struct node* node)
19 {
20     if (node == NULL)
21         return;
22     printPostorder(node->left);
23     printPostorder(node->right);
24     printf("%d ", node->data);
25 }
```



The screenshot shows the Dev-C++ IDE with the file 'tree tran.cpp' open, displaying the 'main' function and the 'printInorder' and 'printPreorder' functions. The 'main' function creates a binary tree with 5 nodes and prints the preorder traversal. A terminal window is open on the right, showing the output of the program.

```
27 void printInorder(struct node* node)
28 {
29     if (node == NULL)
30         return;
31     printInorder(node->left);
32     printf("%d ", node->data);
33     printInorder(node->right);
34 }
35 void printPreorder(struct node* node)
36 {
37     if (node == NULL)
38         return;
39     printf("%d ", node->data);
40     printPreorder(node->left);
41     printPreorder(node->right);
42 }
43 int main()
44 {
45     struct node *root = newNode(1);
46     root->left = newNode(2);
47     root->right = newNode(3);
48     root->left->left = newNode(4);
49     root->left->right = newNode(5);
50
51     printf("\nPreorder traversal of binary tree is \n");
52     printPreorder(root);
53 }
```

Preorder traversal of binary tree is  
1 2 4 5 3  
Inorder traversal of binary tree is  
4 2 5 1 3  
Postorder traversal of binary tree is  
4 5 2 3 1

```
54     printf("\nInorder traversal of binary tree is \n");
55     printInorder(root);
56
57     printf("\nPostorder traversal of binary tree is \n");
58     printPostorder(root);
59
60     getchar();
61     return 0;
```

Compiler Resources Compile Log Debug Find Results

Line: 62 Col: 2 Sel: 0 Lines: 62 Length: 1469 Insert Done parsing in 0.015 seconds

94°F Mostly sunny

13:46 21-09-2022

## EXPERIMENT:-16(STACK APPLICATIONS)

C:\Users\chall\OneDrive\Documents\tree tran.cpp - [Executing] - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

tree tran.cpp

```
1 #include <stdio.h>
2 void TOH(int n, char x, char y, char z){
3     if(n>0){
4         TOH(n-1,x,z,y);
5         printf("\n%c to %c",x,y);
6         TOH(n-1,z,y,x);
7     }
8 }
9 int main()
10 {
11     int n=3;
12     TOH(n,'A','B','C');
13 }
```

Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\chall\OneDrive\Documents\tree tran.exe
- Output Size: 128.4599609375 KiB
- Compilation Time: 0.31s

Process exited after 0.0722 seconds with return value 0  
Press any key to continue . . .

A to B  
A to C  
B to C  
A to B  
C to A  
C to B  
A to B

Line: 7 Col: 6 Sel: 0 Lines: 13 Length: 195 Insert Done parsing in 0 seconds

92°F Mostly sunny

14:52 21-09-2022