Parshvanath Charitable Trust's
A. P. SHAH INSTITUTE OF TECHNOLOGY
(Approved by AICTE New Delhi & Govt. of Maharashtra, Affiliated to University of Mumbai)
(Religious Jain Minority)

Department of Information Technology

Academic Year: 2019-20
Semester: III
Class / Branch: SE IT
Subject: DSL
Name of Instructor:Anagha Aher

Name of Student:Siddhesh Gaikwad
Student ID:18104069
Date of Performance:-14/9/19
Date of Submission:-14/9/19

Experiment No. 07

Aim:-Implementation of BST creation and traversal.

```
#include<stdio.h>
#include<stdlib.h>

typedef struct BST {
   int data;
   struct BST *lchild ,*rchild;
}node;

void insert(node *, node *);
void inorder(node *);
```

```
void preorder(node *);
void postorder(node *);
node *search(node *, int, node **);

void main()
{
        int choice;
        char ans ='N';
        int key;
        node *new_node, *root,*tmp,*parent;
        node *get_node();
        root = NULL;

        printf("\nProgram For Binary Search Tree");
        do {
          printf("\n1.Create");
          printf("\n2.Search");
          printf("\n3.Recursive Traversals");
          printf("\n4.Exit");
          printf("\nEnter your choice :");
          scanf("%d", &choice);

          switch(choice)
```

```
{
    case 1: new_node = get_node();
            printf("\nEnter The Element ");
            scanf("%d", &new_node->data);
            if (root == NULL)
               root = new_node;
            else
               insert(root, new_node);
    break;

    case 2: printf("\nEnter Element to be searched :");
            scanf("%d", &key);

            tmp = search(root, key, &parent);
            printf("\nParent of node %d is %d", tmp->data, parent->data);
    break;

    case 3: if (root == NULL)
               printf("Tree Is Not Created");
            else {
               printf("\nThe Inorder display : ");
               inorder(root);
               printf("\nThe Preorder display : ");
```

```
                        preorder(root);
                        printf("\nThe Postorder display : ");
                        postorder(root);
                         }
                break;
                }
          } while (choice != 4);
}

        node *get_node()
{
        node *temp;
        temp = (node *) malloc(sizeof(node));
        temp->lchild = NULL;
        temp->rchild = NULL;
        return temp;
}


void insert(node *root, node *new_node)
{
        if (new_node->data < root->data)
        {
```

```
        if(root->lchild==NULL)
                root->lchild = new_node;
        else
                insert(root->lchild, new_node);
        }

        if (new_node->data > root->data)
        {
                if (root->rchild == NULL)
                root->rchild = new_node;
        else
                insert(root->rchild, new_node);
        }
}

node *search(node *root, int key, node **parent)
{
        node *temp;
        temp = root;
        while (temp != NULL)
        {
                if (temp->data == key)
                {
```

```
                printf("\nThe %d Element is Present", temp->data);
                return temp;
            }
            *parent = temp;

            if (temp->data > key)
                temp = temp->lchild;
            else
                temp = temp->rchild;
        }
        return NULL;
}

void inorder(node *temp)
{
        if (temp != NULL)
        {
            inorder(temp->lchild);
            printf("%d\t", temp->data);
            inorder(temp->rchild);
        }
}
```

```
void preorder(node *temp)
{
	if (temp != NULL)
	{
		printf("%d\t", temp->data);
		preorder(temp->lchild);
		preorder(temp->rchild);
	}
}


void postorder(node *temp)
{
	if (temp != NULL)
	{
		postorder(temp->lchild);
		postorder(temp->rchild);
		printf("%d\t", temp->data);
	}
}
```

Output:-

apsit@apsit-HP-280-G3-MT:~/Downloads$ gcc EXP7.c

apsit@apsit-HP-280-G3-MT:~/Downloads$ gdb a.out

GNU gdb (Ubuntu 8.1-0ubuntu3) 8.1.0.20180409-git

Copyright (C) 2018 Free Software Foundation, Inc.

License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>

This is free software: you are free to change and redistribute it.

There is NO WARRANTY, to the extent permitted by law. Type "show copying"

and "show warranty" for details.

This GDB was configured as "x86_64-linux-gnu".

Type "show configuration" for configuration details.

For bug reporting instructions, please see:

<http://www.gnu.org/software/gdb/bugs/>.

Find the GDB manual and other documentation resources online at:

<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".

Type "apropos word" to search for commands related to "word"...

Reading symbols from a.out...(no debugging symbols found)...done.

(gdb) r

Starting program: /home/apsit/Downloads/a.out

Program For Binary Search Tree

1.Create

2.Search

3.Recursive Traversals

4.Exit

Enter your choice :1

Enter The Element 99

1.Create

2.Search

3.Recursive Traversals

4.Exit

Enter your choice :1

Enter The Element 80

1.Create

2.Search

3.Recursive Traversals

4.Exit

Enter your choice :1

Enter The Element 101

1.Create

2.Search

3.Recursive Traversals

4.Exit

Enter your choice :1

Enter The Element 70

1.Create

2.Search

3.Recursive Traversals

4.Exit

Enter your choice :1

Enter The Element 88

1.Create

2.Search

3.Recursive Traversals

4.Exit

Enter your choice :1

Enter The Element 100

1.Create

2.Search

3.Recursive Traversals

4.Exit

Enter your choice :1

Enter The Element 105

1.Create

2.Search

3.Recursive Traversals

4.Exit

Enter your choice :3

The Inorder display : 70 80 88 99 100 101 105

The Preorder display : 99 80 70 88 101 100 105

The Postorder display : 70 88 80 100 105 101 99

1.Create

2.Search

3.Recursive Traversals

4.Exit

Enter your choice :2

Enter Element to be searched :88

The 88 Element is Present

Parent of node 88 is 80

1.Create

2.Search

3.Recursive Traversals

4.Exit

Enter your choice :