LAB REPORT
DATA STRUCTURES
RUBIANA JOSEPHINE PAUL
1BM19CS208

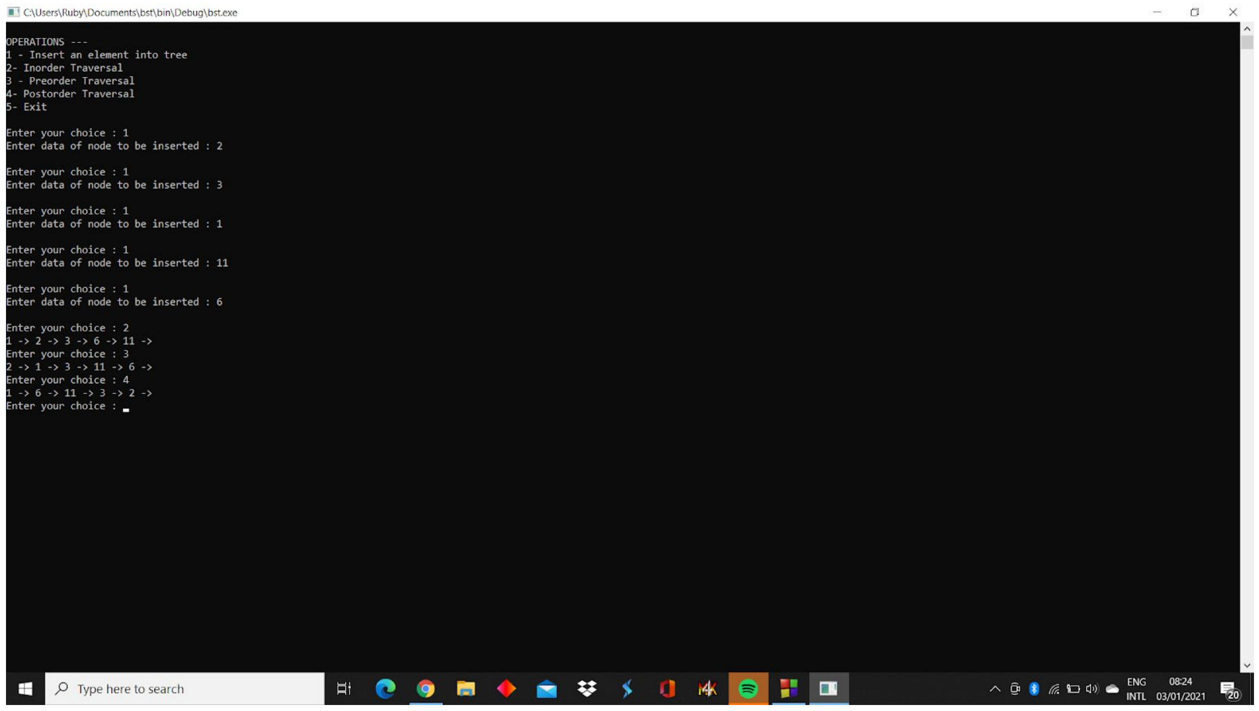
LAB PROGRAM 10
BINARY SEARCH TREE

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

struct btnode
{
   int value;
   struct btnode *l;
   struct btnode *r;
}*root = NULL, *temp = NULL, *t2, *t1;


void insert();

void inorder(struct btnode *t);
void create();
void search(struct btnode *t);

void preorder(struct btnode *t);
void postorder(struct btnode *t);

int flag = 1;

void main()
{
   int ch;

   printf("\nOPERATIONS ---");
   printf("\n1 - Insert an element into tree\n");

   printf("2- Inorder Traversal\n");
   printf("3 - Preorder Traversal\n");
   printf("4- Postorder Traversal\n");
   printf("5- Exit\n");
   while(1)
   {
      printf("\nEnter your choice : ");
```

```
        scanf("%d", &ch);
        switch (ch)
        {
        case 1:
            insert();
            break;
        case 2:
            inorder(root);
            break;
        case 3:
            preorder(root);
            break;
        case 4:
            postorder(root);
            break;
        case 5:
            exit(0);
        default :
            printf("Wrong choice, Please enter correct choice  ");
            break;
        }
    }
}


void insert()
{
    create();
    if (root == NULL)
        root = temp;
        else
        search(root);
}


void create()
{
    int data;

    printf("Enter data of node to be inserted : ");
    scanf("%d", &data);
    temp = (struct btnode *)malloc(1*sizeof(struct btnode));
    temp->value = data;
    temp->l = temp->r = NULL;
```

```
}
 void search(struct btnode *t)
{
    if ((temp->value > t->value) && (t->r != NULL))      /* value more than root node value insert at
right */
        search(t->r);
    else if ((temp->value > t->value) && (t->r == NULL))
        t->r = temp;
    else if ((temp->value < t->value) && (t->l != NULL))    /* value less than root node value insert
at left */
        search(t->l);
    else if ((temp->value < t->value) && (t->l == NULL))
        t->l = temp;
}


void inorder(struct btnode *t)
{
    if (root == NULL)
    {
        printf("No elements in a tree to display");
        return;
    }
    if (t->l != NULL)
        inorder(t->l);
    printf("%d -> ", t->value);
    if (t->r != NULL)
        inorder(t->r);
}



void preorder(struct btnode *t)
{
    if (root == NULL)
    {
        printf("No elements in a tree to display");
        return;
    }
    printf("%d -> ", t->value);
    if (t->l != NULL)
        preorder(t->l);
    if (t->r != NULL)
        preorder(t->r);
```

```
}


void postorder(struct btnode *t)
{
   if (root == NULL)
   {
      printf("No elements in a tree to display ");
      return;
   }
   if (t->l != NULL)
      postorder(t->l);
   if (t->r != NULL)
      postorder(t->r);
   printf("%d -> ", t->value);
}
```



```
OPERATIONS ---
1 - Insert an element into tree
2- Inorder Traversal
3 - Preorder Traversal
4- Postorder Traversal
5- Exit

Enter your choice : 1
Enter data of node to be inserted : 2

Enter your choice : 1
Enter data of node to be inserted : 3

Enter your choice : 1
Enter data of node to be inserted : 1

Enter your choice : 1
Enter data of node to be inserted : 11

Enter your choice : 1
Enter data of node to be inserted : 6

Enter your choice : 2
1 -> 2 -> 3 -> 6 -> 11 ->
Enter your choice : 3
2 -> 1 -> 3 -> 11 -> 6 ->
Enter your choice : 4
1 -> 6 -> 11 -> 3 -> 2 ->
Enter your choice : _
```