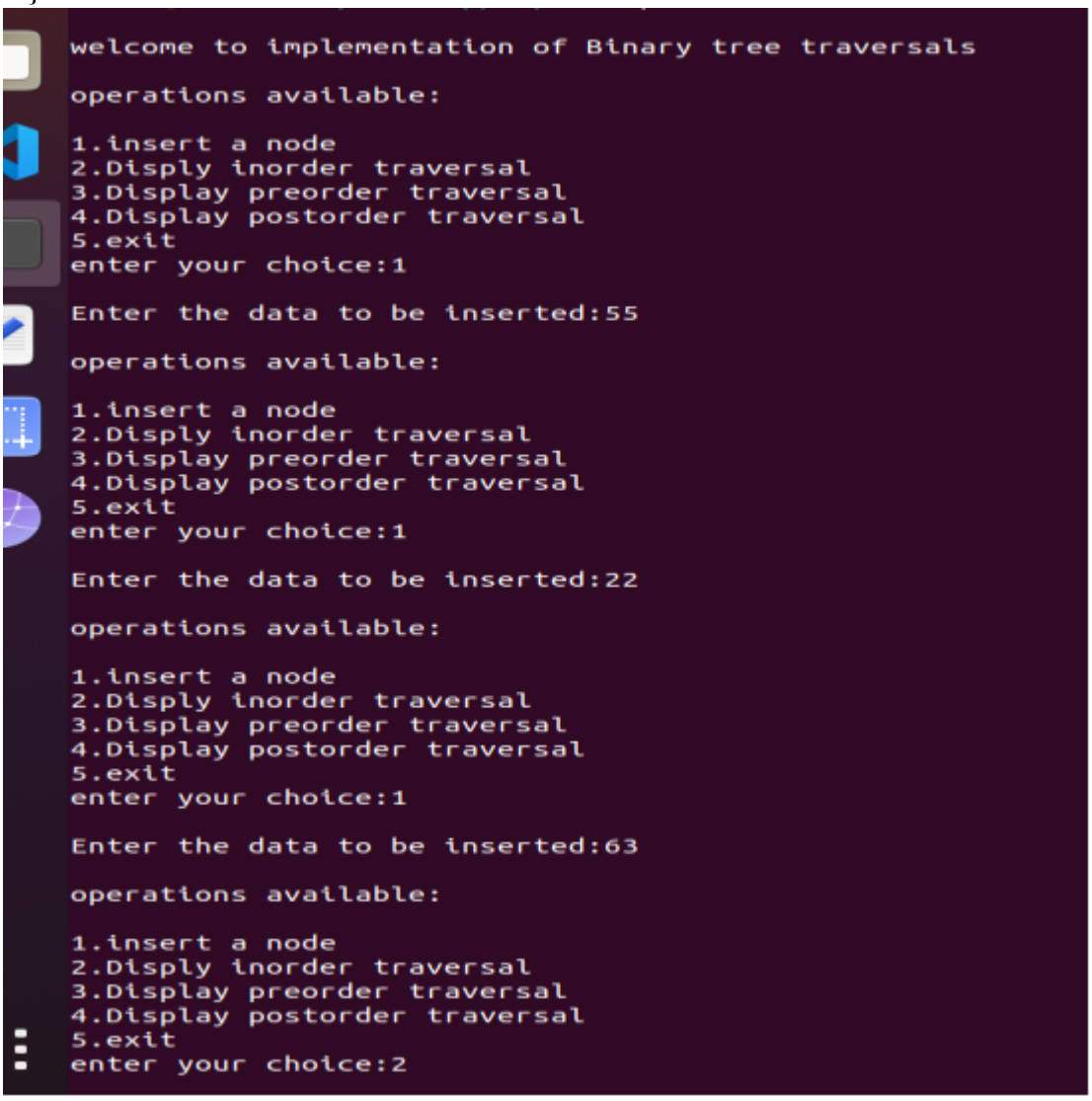
MANALI IT 02:

```
#include<stdio.h>
struct node
{
int data;
struct node *left;
struct node *right;
};
struct node *tree;
void create(struct node *);
struct node *insert(struct node *,int);
void inorder(struct node *);
void preorder(struct node *);
void postorder(struct node *);
void main()
{
printf("\nwelcome to implementation of Binary tree traversals\n");
int choice,x;
struct node *ptr;
create(tree);
do
{
printf("\noperations available:\n");
printf("\n1.insert a node");
printf("\n2.Disply inorder traversal");
printf("\n3.Display preorder traversal");
printf("\n4.Display postorder traversal");
printf("\n5.exit");
printf("\nenter your choice:");
scanf("%d",&choice);
switch(choice)
{
case 1:
printf("\nEnter the data to be inserted:");
scanf("%d",&x);
tree=insert(tree,x);
break;
case 2:
printf("\nelements in the inorder traversal are:");
inorder(tree);
printf("\n");
break;
case 3:
printf("\nelements in the preorder traversal are:");
preorder(tree);
printf("\n");
break;
case 4:
printf("\nelements in the postorder traversal are:");
postorder(tree);
printf("\n");
break;
case 5:
printf("\nExit, Program Finished!!");
break;
```

```
default:
printf("\nplease enter valid option 1,2,3,4,5.");
break;
}
}
while(choice!=5);
}
void create(struct node *tree)
{
tree=NULL;
}
struct node *insert(struct node *tree,int x)
{
  struct node *p,*temp,*root;
  p=(struct node *)malloc(sizeof(struct node));
  p->data=x;
  p->left=NULL;
  p->right=NULL;
  if(tree==NULL)
  {
  tree=p;
  tree->left=NULL;
  tree->right=NULL;
  }
  else
  {
  root=NULL;
  temp=tree;
  while(temp!=NULL)
  {
  root=temp;
  if(x<temp->data)
  {
  temp=temp->left;
  }
  else
  {
  temp=temp->right;
  }
  }
  if(x<root->data)
  {
  root->left=p;
  }
  else
  {
  root->right=p;
  }
  }
  return tree;
  }
  void inorder(struct node *tree)
  {
  if(tree!=NULL)
  {
  inorder(tree->left);
  printf("%d\t",tree->data);
```

```
inorder(tree->right);
}
}
void preorder(struct node *tree)
{
if(tree!=NULL)
{
printf("%d\t",tree->data);
preorder(tree->left);
preorder(tree->right);
}
}
void postorder(struct node *tree)
{
if(tree!=NULL)
{
postorder(tree->left);
postorder(tree->right);
printf("%d\t",tree->data);
}
}
```



```
elements in the inorder traversal are:22        55      63

operations available:

1.insert a node
2.Disply inorder traversal
3.Display preorder traversal
4.Display postorder traversal
5.exit
enter your choice:3

elements in the preorder traversal are:55       22      63

operations available:

1.insert a node
2.Disply inorder traversal
3.Display preorder traversal
4.Display postorder traversal
5.exit
enter your choice:4

elements in the postorder traversal are:22      63      55

operations available:

1.insert a node
2.Disply inorder traversal
3.Display preorder traversal
4.Display postorder traversal
5.exit
enter your choice:5

Exit, Program Finished!!dl408@itadmin:~/Desktop/expt6$
```