```
//Binary Tree


# include <stdio.h>

# include <conio.h>


struct binary

{

        int no;

        struct binary *left, *right;

};


struct binary *root;


void main()

{

        int choice;

        void create();

        void insert();

        void preorder(struct binary *);

        void inorder(struct binary *);

        void postorder(struct binary *);

        void traversing();


        clrscr();

        create();
```

```
do
{
        printf("\n\t1. Insert");
        printf("\n\t2. Traversing");
        printf("\n\t3. PreOrder");
        printf("\n\t4. InOrder");
        printf("\n\t5. PostOrder");
        printf("\n\t0. Exit");

        printf("\n\tEnter your choice : ");
        scanf("%d",&choice);

        switch(choice)
        {
                case 1:
                        insert();
                        break;
                case 2:
                        traversing();
                        break;
                case 3:
                        preorder(root);
                        break;
                case 4:
```

```
                    inorder(root);
                    break;
                case 5:
                    postorder(root);
                    break;
                case 0:
                    printf("\n\tEnd of program");
                    break;
                default:
                    printf("\n\tInvalid Choice");
                    break;
            }
            getch();
        }
        while(choice != 0);
}
void create()
{
        struct binary *parent, *node;
        int info;
        char ans;

        do
        {
            printf("Enter number for node information : ");
```

```
scanf("%d",&info);


if(root == NULL)
{
        node = (struct binary *) malloc(sizeof(struct binary));
        root = node;
}
else
{
        parent = root;


        while(parent != NULL)
        {
                if(info > parent->no)
                {
                        node = parent;
                        parent = parent->right;
                }
                else
                {
                        node = parent;
                        parent = parent->left;
                }
        }
```

```
                        if(info > node->no)
                        {
                                node->right = (struct binary *) malloc(sizeof(struct binary));
                                node = node->right;
                        }
                        else
                        {
                                node->left = (struct binary *) malloc(sizeof(struct binary));
                                node = node->left;
                        }
                }
                node->no = info;
                node->left = NULL;
                node->right = NULL;

                printf("\n\tDo you want to add another node ? : ");
                fflush(stdin);
                scanf("%c", &ans);
        }
        while(ans == 'y' || ans == 'Y');
}

void insert()
{
        struct binary *parent, *node;
```

```
int info;


printf("\n\tEnter any number for new node : ");
scanf("%d",&info);


if(root == NULL)
{
        node = (struct binary *)malloc(sizeof(struct binary));
        root = node;
}
else
{
        parent = root;


        while(parent != NULL)
        {
                if(info > parent->no)
                {
                        node = parent;
                        parent = parent->right;
                }
                else
                {
                        node = parent;
                        parent = parent->left;
```

```
                }
        }

        if(info > node->no)
        {
                node->right = (struct binary *) malloc(sizeof(struct binary));
                node = node->right;
        }
        else
        {
                node->left = (struct binary *) malloc(sizeof(struct binary));
                node = node->left;
        }
    }
    node->no = info;
    node->left = NULL;
    node->right=NULL;



}

void preorder(struct binary *temp)
{
    if(temp)
    {
```

```
                printf("%d, ", temp->no);
                preorder(temp->left);
                preorder(temp->right);
        }

}
void inorder(struct binary *temp)
{
        if(temp)
        {
                inorder(temp->left);
                printf("%d, ", temp->no);
                inorder(temp->right);
        }
}

void postorder(struct binary *temp)
{
        if(temp)
        {
                postorder(temp->left);
                postorder(temp->right);
                printf("%d, ", temp->no);
        }
}
```

```
void traversing()
{
	printf("\n\tPre Order");
	preorder(root);

	printf("\n\tIn Order");
	inorder(root);

	printf("\n\tPost Order");
	postorder(root);

}
```