

## Trees

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

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

```
typedef struct node {
```

```
    int data;
```

```
    struct node *left, *right;
```

```
};
```

```
node *create(int data) {
```

```
    node *temp;
```

```
    temp = (node *) malloc (sizeof(node));
```

```
    temp->data = data;
```

```
    temp->left = temp->right = NULL;
```

```
    return temp;
```

```
}
```

```
void inorder(node *root) {
```

```
    if (root != NULL) {
```

```
        inorder(root->left);
```

```
        printf("%d", root->data);
```

```
        inorder(root->right);
```

```
}
```

```
}
```



```

void preorder (node * root) {
    if (root != NULL) {
        printf("%d", root->data);
        preorder (root->left);
        preorder (root->right);
    }
}

```

```

void postorder (node * root) {
    if (root != NULL) {
        postorder (root->left);
        postorder (root->right);
        printf("%d", root->data);
    }
}

```

```

void insert (node * root, node * temp) {
    if (temp->data < root->data) {
        if (root->left == NULL)
            insert (root->left, temp);
        else
            root->left = temp;
    }
}

```



```
int main(void) {  
    node *not = NULL, *temp;
```

```
    int choice = 0;  
    while (choice != 2)  
    {
```

```
        temp =
```

```
        printf("1 - insert\n");
```

```
        printf("2 - Exit\n");
```

```
        printf("Enter your choice: ");
```

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

```
        if (choice == 1)
```

```
        {
```

```
            int val;
```

```
            printf("Enter Value: ");
```

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

```
            temp = create(val);
```

```
            if (not == NULL)
```

```
                not = temp;
```

```
            else
```

```
                insert(not, temp);
```



else if (choice == 2)

break;

3

Printk ("Inorder");

inorder(root);

Printk ("preorder");

preorder(root);

Printk ("postorder");

postorder(root);