LINKED LIST

```c
#include<stdio.h>
#include<stdlib.h>
struct Node{
    int data;
    struct Node *next;
};

int main()
{
    struct Node *head = (struct Node *)malloc(sizeof(struct Node));
    struct Node *ptr_2 = (struct Node *)malloc(sizeof(struct Node));
    struct Node *ptr_3 = (struct Node *)malloc(sizeof(struct Node));
    // Inserting the data;
    head->data = 56;
    ptr_2->data = 34;
    ptr_3->data = 98;
    // Connecting the node;
    head->next = ptr_2;
    ptr_2->next = ptr_3;
    ptr_3->next = NULL;
```

Insertion

```c
struct Node *Insert_At_head(struct Node *head, int data){
    struct Node *ptr=(struct Node *)malloc(sizeof(struct Node));
    ptr->data = data;
    ptr->next = head;
    return ptr;
}
struct Node *Insert_At_End(struct Node *node, int data){
    struct Node *ptr=(struct Node *)malloc(sizeof(struct Node));
    struct Node *p=node;
    ptr->data = data;
    while(p->next!=NULL){
        p=p->next;}
    p->next=ptr;
    ptr->next=NULL;
    return node;
}
struct Node *Insert_At_Pos(struct Node *node, int data,int pos){
    struct Node *new=(struct Node *)malloc(sizeof(struct Node));
    struct Node *prev=node;
    new->data = data;
```

```
    int i=0;
    while(i<pos-1 && prev->next!=NULL){
        prev=prev->next;
        i++;
    }
    new->next=prev->next;
    prev->next=new;
    return node;
}
```

Binary Tree

```
#include <stdio.h>
#include <stdlib.h>
struct Node{
    int data;
    struct Node *left;
    struct Node *right;
};
struct Node *createNode(int data){
    struct Node *n = (struct Node *)malloc(sizeof(struct Node));
    n->data = data;
    n->left = NULL;
    n->right = NULL;
    return n;
}
void Order(struct Node *root){
    if (root != NULL){
        //if pre->1 ,in->2 ,post->3//"Printf" POSITION
        printf("%d ", root->data);
        Order(root->left);
        Order(root->right);
    }
}
int main(){
    struct Node *p = createNode(5);
    struct Node *p1 = createNode(3);
    struct Node *p2 = createNode(6);
    p->left = p1;
    p->right = p2;}
```