Manoj M Mallya

200905130

Section : C2

Roll no. : 23

DSAL – 8 - TREE CONCEPTS

Solved example :

1) Create a binary tree using recursion and display its elements using all the traversal methods.

Code :

```
#include<stdio.h>

typedef struct node
{
 int data;
 struct node *lchild;
 struct node *rchild;
 }*NODE;

NODE Create_Binary_Tree()
{
 NODE temp;
 int ele;
 printf("Enter the element to inserted (-1 for no data):");
```

```
 scanf("%d",&ele);
 if(ele==-1)
 return NULL;
 temp=(NODE*)malloc(sizeof(struct node));
 temp->data=ele;
 printf("Enter lchild child of %d:\n",ele);
 temp->lchild=Create_Binary_Tree();
 printf("Enter rchild child of %d:\n",ele);
 temp->rchild=Create_Binary_Tree();
 return temp;
}

void inorder(NODE ptr)
{
if(ptr!=NULL)
{
inorder(ptr->lchild);
printf("%5d",ptr->data);
inorder(ptr->rchild);
}
}

void postorder(NODE ptr)
{
if(ptr!=NULL)
{
postorder(ptr->lchild);
```

```
postorder(ptr->rchild);
printf("%5d",ptr->data);
}
}

void preorder(NODE ptr)
{
if(ptr!=NULL)
{
printf("%5d",ptr->data);
preorder(ptr->lchild);
preorder(ptr->rchild);
}
}

int main()
{
printf("Manoj M Mallya\n200905130\nSection : C2\nRoll no : 23\n\n");
int n,ch,i;
NODE *root;
root=NULL;
while(1)
{
printf("**********Output**********\n\n");
printf("----------Menu----------\n");
printf(" 1. Insert\n 2. All traversals\n 3. Exit\n");
printf("Enter your choice:");
```

```
scanf("%d",&ch);
switch(ch)
{

case 1: printf("Enter node :\n");
root=Create_Binary_Tree();
break;

case 2: printf("\nInorder traversal:\n");
inorder(root);
printf("\nPreorder traversal:\n");
preorder(root);
printf("\nPostorder traversal:\n");
postorder(root);
printf("\n\n**********************************************");
break;

case 3: exit(0);
}
}

return 0;
}
```

Output :

```
"D:\manoj MIT\code blocks programs\C\binary tree\bin\Debug\binary tree.exe"
Manoj M Mallya
200905130
Section : C2
Roll no : 23

**********Output**********

----------Menu----------
 1. Insert
 2. All traversals
 3. Exit
Enter your choice:1
Enter node :
Enter the element to inserted (-1 for no data):10
Enter lchild child of 10:
Enter the element to inserted (-1 for no data):20
Enter lchild child of 20:
Enter the element to inserted (-1 for no data):-1
Enter rchild child of 20:
Enter the element to inserted (-1 for no data):30
Enter lchild child of 30:
Enter the element to inserted (-1 for no data):-1
Enter rchild child of 30:
Enter the element to inserted (-1 for no data):-1
Enter rchild child of 10:
Enter the element to inserted (-1 for no data):40
Enter lchild child of 40:
Enter the element to inserted (-1 for no data):-1
Enter rchild child of 40:
Enter the element to inserted (-1 for no data):-1
**********Output**********
```

```
**********Output**********

----------Menu----------
 1. Insert
 2. All traversals
 3. Exit
Enter your choice:2

Inorder traversal:
   20   30   10   40
Preorder traversal:
   10   20   30   40
Postorder traversal:
   30   20   40   10

**********Output**********

----------Menu----------
 1. Insert
 2. All traversals
 3. Exit
Enter your choice:3

Process returned 0 (0x0)   execution time : 39.350 s
Press any key to continue.
```

Questions for Lab8 :

1) Add two long positive integers represented using circular doubly linked list with header node.

Code :

```
#include<stdio.h>
#include<stdlib.h>
typedef struct node
{
   int data;
   struct node * next;
   struct node * prev;
} * NODE;
void insertFront(NODE head, int val)
{
   NODE first = head->next;
   head->data++;
   NODE n = (NODE)malloc(sizeof(struct node));
   n->data = val;
   n->next = NULL;
   n->prev = NULL;
   if(first == NULL)
   {
      n->next = n;
      n->prev = n;
      head->next = n;
      return;
   }
   n->next = first;
```

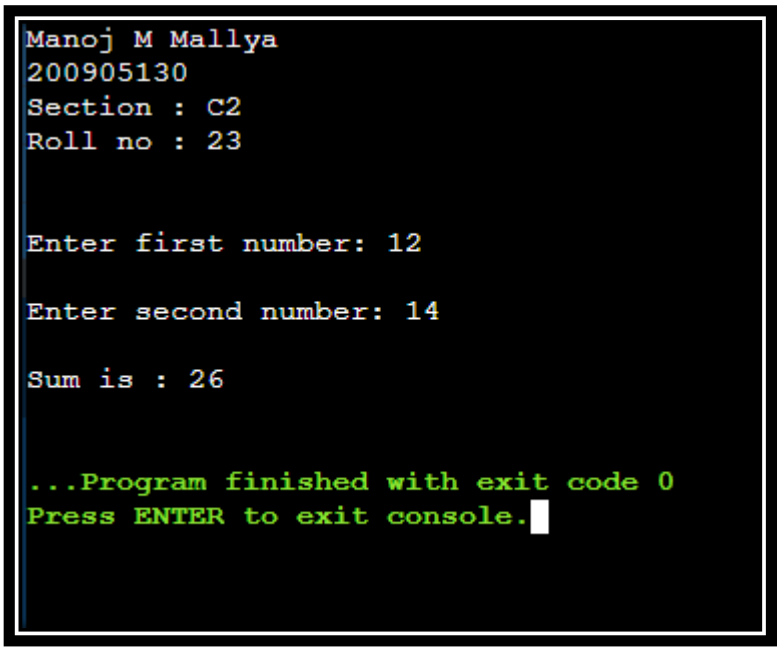
```
    n->prev = first->prev;
    (first->prev)->next = n;
    first->prev = n;
    head->next = n;
    return;
}
void display(NODE head)
{
    if(head->data == 0)
    {
        printf("\nList is empty\n");
        return;
    }
    NODE first = head->next;
    NODE temp = first;
    while(temp->next!=first)
    {
        printf("%d",temp->data);
        temp = temp->next;
    }
    printf("%d\n",temp->data);
}
void insert(NODE head, int val)
{
    int x;
    while(val>0)
    {
```

```
        x=val%10;
        val/=10;
        insertFront(head,x);
    }
}
NODE addLists(NODE head1, NODE head2)
{
    if(head1->data == 0)
    {
        return head2;
    }
    if(head2->data == 0)
    {
        return head1;
    }
    int c1 = head1->data;
    int c2 = head2->data;
    int diff = c1-c2;
    int s = 0,i;
    if(diff < 0)
    {
        diff *= -1;
        for(i=0; i<diff; ++i)
        {
            insertFront(head1,0);
        }
    }
```

```
else if(diff > 0)
{
   for(i = 0; i<diff; ++i)
   {
      insertFront(head2,0);
   }
}
NODE sum = (NODE)malloc(sizeof(struct node));
sum->data = 0;
int carry = 0;
NODE f1 = head1->next;
NODE f2 = head2->next;
NODE op1 = f1->prev;
NODE op2 = f2->prev;
while(op1!=f1 && op2!=f2)
{
   s = (op1->data)+(op2->data)+carry;
   carry = s/10;
   s%=10;
   insertFront(sum,s);
   op1 = op1->prev;
   op2 = op2->prev;
}
s = (op1->data)+(op2->data)+carry;
carry = s/10;
s%=10;
insertFront(sum,s);
```

```
    if(carry != 0)
    {
        insertFront(sum,carry);
    }
    return sum;
}
int main()
{
    printf("Manoj M Mallya\n200905130\nSection : C2\nRoll no : 23\n\n");
    NODE head1 = (NODE)malloc(sizeof(struct node));
    NODE head2 = (NODE)malloc(sizeof(struct node));
    NODE sum = NULL;
    int v1,v2;
    head1->data = 0;
    head2->data = 0;
    printf("\nEnter first number: ");
    scanf("%d",&v1);
    insert(head1,v1);
    printf("\nEnter second number: ");
    scanf("%d",&v2);
    insert(head2,v2);
    sum = addLists(head1,head2);
    printf("\nSum is : ");
    display(sum);
    return 0;
}
```

Output :



2) Write a menu driven program to do the following using iterative functions:

i) To create a BST for a given set of integer numbers

ii) To delete a given element from BST.

iii) Display the elements using iterative in-order traversal.

Code :

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 10
typedef struct node{
int key;
struct node *left, *right;
}* NODE;
typedef struct{
NODE S[MAX];
```

```
int tos;
}STACK;
NODE newNODE (int item){
NODE temp = (NODE)malloc(sizeof(struct node));
temp->key = item;
temp->left = temp->right = NULL;
return temp;
}
void push (STACK *s, NODE n){
s->S[++(s->tos)] = n;
}
NODE pop (STACK *s){
return s->S[(s->tos)--];
}
void inorder (NODE root){
NODE curr;
curr = root;
STACK S;
S.tos = -1;
push(&S, root);
curr = curr->left;
while (S.tos != -1 || curr != NULL){
while (curr != NULL){
push(&S, curr);
curr = curr->left;
}
curr = pop(&S);
```

```
printf("%d\t", curr->key);
curr = curr->right;
}}
NODE insert (NODE node, int key){
if (node == NULL)
return newNODE(key);
if (key < node->key)
node->left = insert(node->left, key);
else if (key > node->key)
 node->right = insert(node->right, key);
 return node;
}
NODE minValueNode (NODE node){
NODE current = node;
while (current && current->left != NULL)
current = current->left;
return current;
}
NODE deleteNode (NODE root, int key){
if (root == NULL)
return root;
if (key < root->key)
root->left = deleteNode(root->left, key);
else if (key > root->key)
root->right = deleteNode(root->right, key);
else{
if (root->left == NULL){
```

```
NODE temp = root->right;
free(root);
return temp;
}
else if (root->right == NULL){
NODE temp = root->left;
free(root);
return temp;
}
NODE temp = minValueNode(root->right);
root->key = temp->key;
root->right = deleteNode(root->right, temp->key);
}
return root;
}
int main(){
   printf("Manoj M Mallya\n200905130\nSection : C2\nRoll no : 23\n\n");
NODE root = NULL;
int k;
printf("Enter the root:\t");
scanf("%d", &k);
root = insert(root, k);
int ch;
while(1){
printf("\n1. Insert\n2. Delete\n3. Display\n4. Exit:\n");
printf("Enter your choice : ");
scanf("%d", &ch);
```

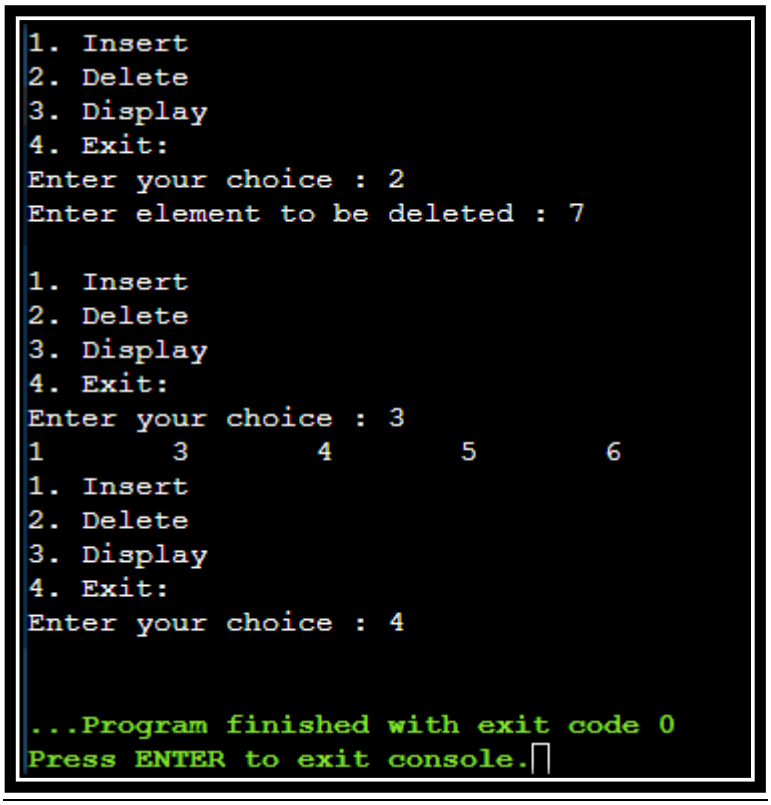
```
switch (ch){
 case 1: printf("Enter element to be inserted : ");
 scanf("%d", &k);
 root = insert(root, k);
 break;
 case 2: printf("Enter element to be deleted : ");
 scanf("%d", &k);
 root = deleteNode(root, k);
 break;
 case 3: inorder(root);
 break;
 case 4: return 0;
}}}
```

Output :

```
Manoj M Mallya
200905130
Section : C2
Roll no : 23

Enter the root: 7

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 1
Enter element to be inserted : 1

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 1
Enter element to be inserted : 6
```

```
Enter your choice : 1
Enter element to be inserted : 6

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 1
Enter element to be inserted : 2

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 1
Enter element to be inserted : 5

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 1
Enter element to be inserted : 3

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 1
Enter element to be inserted : 4

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 3
1       2       3       4       5       6       7
```

```
Enter your choice : 3
1       2       3       4       5       6       7
1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 2
Enter element to be deleted : 2

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 2
```


1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 2
Enter element to be deleted : 7
1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 3
1 3 4 5 6
1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 4
...Program finished with exit code 0
Press ENTER to exit console.

****