```
#include<stdio.h>
#include<ctype.h>
#include<stdlib.h>
#include<string.h>
#include<math.h>
#include<time.h>
#include<conio.h>
#include<Windows.h>
#include<WindowsX.h>

#define LED_PIN 13;
#define BinTree struct binTree{};

//Today is Binary Tree

struct data{
    int angka;

    data *left;
    data *right; //recursive!!!
}*root = NULL;

void insertNode(data **temp, int angka){
    if(*temp==NULL){
        *temp = (data *) malloc(sizeof(data));
        (*temp)->angka = angka;
        (*temp)->left = NULL;
        (*temp)->right = NULL;
    }
    else if(angka > (*temp)->angka) insertNode(&(*temp)->right, angka);
    else if(angka < (*temp)->angka) insertNode(&(*temp)->left, angka);

}

data **getAnakKiriPalingKanan(data**temp){
    if((*temp)->right!=NULL){
        getAnakKiriPalingKanan(&(*temp)->right);
    }
    else{
        return temp;
    }
} //lecturer's


void popPengganti(data **temp){
    if((*temp)->left==NULL && (*temp)->right==NULL){
        //if leaf
        free(*temp);
        *temp=NULL;
    }
    else if((*temp)->left!=NULL && (*temp)->right==NULL){
        //if there is left child, there is no right child, right filled!
        data *temp2 = *temp;
        *temp = (*temp)->left;
        free(temp2);
    }
```

```
57      else if((*temp)->right!=NULL && (*temp)->left==NULL){
58          //if there is no left child, there is right child, left filled!
59          data *temp2 = *temp;
60          *temp = (*temp)->right;
61          free(temp2);
62      }
63      else if((*temp)->right!=NULL && (*temp)->left!=NULL){
64          //if there is left child, there is right child, both filled!
65          data **curr = getAnakKiriPalingKanan(&(*temp)->left);
66          (*temp)->angka = (*curr)->angka;
67          popPengganti(curr); //get into this node's family
68      }
69  } //lecturer's
70
71
72  void pop(data **curr,int angka){
73      if((*curr)->angka == angka){
74          popPengganti(curr);
75      }
76      else{
77          if(angka>(*curr)->angka)pop(&(*curr)->right,angka);
78          else if(angka<(*curr)->angka)pop(&(*curr)->left,angka);
79      }
80  } //lecturer's
81
82  void eraseNode(data **temp, int angka){ //pop(&root, 10);
83      /* Conditions
84      if leaf
85      if 1 child
86      if 2 children
87      */
88      if(*temp==NULL){
89
90      }
91  }
92
93  void preOrder(data **temp){
94      if(*temp!=NULL){
95          printf("%d ",(*temp)->angka);
96          preOrder(&(*temp)->left);
97          preOrder(&(*temp)->right);
98      }
99  }
100
101 void inOrder(data **temp){
102     if(*temp!=NULL){
103         inOrder(&(*temp)->left);
104         printf("%d ",(*temp)->angka);
105         inOrder(&(*temp)->right);
106     }
107 }
108
109 void postOrder(data **temp){
110     if(*temp!=NULL){
111         postOrder(&(*temp)->left);
112         postOrder(&(*temp)->right);
```

```
113         printf("%d ",(*temp)->angka);
114     }
115 }
116 
117 void printDatas(int mode){
118     data **nd = &root;
119 
120     if(mode == 0){
121         /*printf("%d", (*nd)->angka);
122         *nd = (*nd)->left;
123         if((*nd) != NULL){
124             printDatas(0);
125         }*/
126     } else if(mode == 1){
127 
128     } else if(mode == 2){
129 
130     }
131 }
132 
133 int main(){
134     int select = 0, kounter = 0;
135 
136     ////hmst;
137     //printf("Press Enter to clrscr in Visual!");
138     //getchar();
139 
140     ////printf("\033[0J");
141     ////printf("%c", 12);
142     //system("cls"); //bad idea!
143     //system("color fc"); // here's why
144     //printf("Please do not use system();!\n dangerous!");
145     //getchar();
146     //system("color");
147 
148     /*insertNode(&root,13);
149     insertNode(&root,15);
150     insertNode(&root,10);*/
151 
152     insertNode(&root,15);
153     insertNode(&root,7);
154     insertNode(&root,9);
155     insertNode(&root,8);
156 
157     pop(&root,9);
158     printf("Pre-order: ");
159     //printDatas(0);
160     preOrder(&root);
161     printf("\n");
162 
163     printf("In-order: ");
164     //printDatas(1);
165     inOrder(&root);
166     printf("\n");
167 
168     printf("Post-order: ");
```

```
169     //printDatas(2);
170     postOrder(&root);
171     printf("\n");
172
173     getchar();
174     return 0;
175 }
176
177 //HOMEwork!
178 /*
179 Binary Tree, football club player list
180
181 view menu -> preOrder
182 exit and remove all -> popALL
183 add -> Decide between left or right of previous trees. I mean if tree is not
      NULL
184 */
```