```c
#include "esercizio.h"
#include <stdio.h>
void nodiNormaliAux(TipoAlbero a, int l, Insieme* r){
        //Implementare
}

Insieme* nodiNormali(TipoAlbero a, int l){
if(a==NULL) return insiemeVuoto();
if(l!=0){
        Insieme *sinist = nodiNormali(a->sinistro,l-1);
        Insieme *dest   = nodiNormali(a->destro,l-1  );
  IteratoreInsieme* it = creaIteratoreInsieme(dest);
        if(hasNext(it)){
                inserisci(sinist,next(it));
        }
}
if(l==0){if(estVuoto(a->sinistro)==estVuoto(a->destro)){
        Insieme *def = insiemeVuoto();
        inserisci(def,a->info);
        return def;
}}
return insiemeVuoto();
}

bool estFoglia(TipoAlbero a){// Funzione ausiliaria di tagliaFoglie
        if(destro(a)==NULL && sinistro(a)==NULL )return true;
        return false;
}

void tagliaFoglie(TipoAlbero* a){
if((*a)!=NULL){

        if( estVuoto(sinistro(*a)) && !estVuoto(destro(*a))){
                if(estFoglia(((*a)->destro))){
                        (*a)->info+=(*a)->destro->info;
                        (*a)->destro=NULL;
                }
        }

        if(!estVuoto(sinistro(*a)) &&  estVuoto(destro(*a))){
                if(estFoglia(((*a)->destro))){
                        (*a)->info+=(*a)->sinistro->info;
                        (*a)->sinistro= NULL;
                }
        }
        tagliaFoglie(&((*a)->sinistro));
        tagliaFoglie(&((*a)->destro));
        }
return;

}

void normalizza(TipoAlbero* a){
if(!estVuoto(*a)){
        if( estVuoto(sinistro(*a)) && !estVuoto(destro(*a))){
        TipoAlbero beb = creaAlbBin((*a)->destro->info,NULL,NULL);
        (*a)->sinistro= beb;
        }
        if(!estVuoto(sinistro(*a)) &&  estVuoto(destro(*a))){
        TipoAlbero beb = creaAlbBin((*a)->sinistro->info,NULL,NULL);
        (*a)->destro=beb;
        }
        normalizza(&((*a)->sinistro));
        normalizza(&((*a)->destro  ));
}
return;
}
```

```
TipoAlbero normalizzaFunzionale(TipoAlbero a){
if(a==NULL) return NULL;

if( estVuoto(sinistro(a)) && !estVuoto(destro(a))){
        TipoNodoAlbero* ciao = nodoalb_alloc(a->destro->info);
        a->sinistro = ciao;
}

if(!estVuoto(sinistro(a)) &&  estVuoto(destro(a))){
        TipoNodoAlbero* ciao = nodoalb_alloc(a->sinistro->info);
        a->destro = ciao;
}
TipoAlbero heh = creaAlbBin(a->info,a->sinistro,a->destro);
heh->sinistro=normalizzaFunzionale(a->sinistro);
heh->destro=normalizzaFunzionale(a->destro);
return heh;







}


void scambiaSCL(TipoLista* l){
if((*l) != NULL){
        if((*l)->next!=NULL){
                if((*l)->info>(*l)->next->info){
                        TipoInfoLista ciao = (*l)->info;
                        (*l)->info=(*l)->next->info;
                        (*l)->next->info=ciao;
                }
                scambiaSCL(&((*l)->next));
        }
}
else return;
}


TipoAlbero tagliaFoglieFunzionale(TipoAlbero a){
if(a==NULL) return NULL;
        if( estVuoto(sinistro(a)) && !estVuoto(destro(a))){
                TipoNodoAlbero* ciao = nodoalb_alloc(a->info);
                ciao->info= a->info+a->destro->info;
                return ciao;
        }
        if(!estVuoto(sinistro(a)) &&  estVuoto(destro(a))){
                TipoNodoAlbero* ciao = nodoalb_alloc(a->info);
                ciao->info= a->info+a->sinistro->info;
                return ciao;
        }
TipoAlbero buon = creaAlbBin(a->info,a->sinistro,a->destro);
buon->sinistro=tagliaFoglieFunzionale(a->sinistro);
buon->destro  =tagliaFoglieFunzionale(a->destro  );
return buon;
}
```