

Un exemple del funcionament de l'operació `afegir` a partir d'una llista buida 1:

```
l.afegir(10); l = [10, 1]
l.afegir(20); l = [10, 1] [20, 1]
l.afegir(10); l = [10, 2] [20, 1]
l.afegir(20); l = [10, 2] [20, 2]
l.afegir(20); l = [20, 3] [10, 2]
l.afegir(10); l = [20, 3] [10, 3]
l.afegir(30); l = [20, 3] [10, 3] [30, 1]
l.afegir(30); l = [20, 3] [10, 3] [30, 2]
l.afegir(30); l = [20, 3] [10, 3] [30, 3]
l.afegir(30); l = [30, 4] [20, 3] [10, 3]
```

Feu servir la següent especificació:

```
// Pre: cert
// Post: Si valor no hi és, s'afegeix amb freqüència 1,
// si hi és, se li suma 1 a la freqüència.
void afegir(const T& valor);
```

No fer servir operacions públiques de la classe `Llista`.

SOLUCIÓ:

Posar el nom i contestar al full de resposta.


```

else{ // valor sí hi era
    [ ] ;
    node_llista* aux = n->ant; // cercar on posar el node
    while( [ ] ){
        aux = aux->ant;
    }
    if (aux != n->ant){ // cal fer canvis
        if (n == ultim_node){
            [ ] ;
            [ ] ;
        }
        else{
            [ ] ;
            [ ] ;
        }
        if (aux == nullptr){
            [ ] ;
            [ ] ;
            [ ] ;
            [ ] ;
        }
        else{
            [ ] ;
            [ ] ;
            [ ] ;
            [ ] ;
        }
    }
}
}
}
}

```


Implementeu els següents mètodes:

```
// Pre:  $k \geq 0$ , la mida de  $v[i..j]$  és  $2^k$ ,  $0 \leq i \leq j < v.size()$ ,  
// tots els camins de la jerarquia de nodes a partir d'n  
// tenen al menys longitud  $k+1$ ,  
// tots els nodes de la jerarquia de nodes a partir d'n  
// tenen o zero o dos successors  
// Post: la jerarquia de nodes a partir d'n  
// està normalitzada amb  $k$  i  $v[i..j]$   
static void normalitza_rec(node_arbre* n, int k,  
                           const vector<T>& v, int i, int j);  
  
// Pre:  $k \geq 0$ , la mida de  $v$  és  $2^k$ ,  
// tots els camins del p.i. tenen al menys longitud  $k+1$ ,  
// tots els nodes del p.i. tenen o zero o dos successors  
// Post: el p.i. està normalitzat amb  $k$  i  $v$   
void normalitza(int k, const vector<T>& v);
```

No fer servir operacions públiques de la classe `Arbre`.

SOLUCIÓ:

Posar el nom i contestar al full de resposta.

