```
#include
#include
#include
#include

using namespace std;

template
void preorden(const ArbolBinario& a,
const typename ArbolBinario::Nodo n){
if (n!=0){
cout << a.etiqueta(n) << ' ';
preorden(a,a.izquierda(n));
preorden(a,a.derecha(n));
}
}

template
void inorden(const ArbolBinario& a,
const typename ArbolBinario::Nodo n){
if (n!=0){
inorden(a,a.izquierda(n));
cout << a.etiqueta(n) << ' ';
inorden(a,a.derecha(n));
}
}

template
void postorden(const ArbolBinario& a,
const typename ArbolBinario::Nodo n){
if (n!=0){
postorden(a,a.izquierda(n));
postorden(a,a.derecha(n));
cout << a.etiqueta(n) << ' ';
}
}

template
int Altura(const ArbolBinario& a,
const typename ArbolBinario::Nodo n){
int iz,de;
if (n==0)
```

```
return -1;
else {
iz= Altura(a,a.izquierda(n));
de= Altura(a,a.derecha(n));
return 1+(iz>de?iz:de);
}
}

template
void refleja(ArbolBinario& a){
ArbolBinario ai,ad;

if (!a.empty()){
a.podar_izquierda(a.raiz(),ai);
a.podar_derecha(a.raiz(),ad);
refleja(ai);
refleja(ad);
a.insertar_izquierda(a.raiz(),ad);
a.insertar_derecha(a.raiz(),ai);
}
}

template
void Esquema(const ArbolBinario& a,
const typename ArbolBinario::Nodo n, string& pre){
int i;

if (n==0)
cout << pre << "-- x" << endl;
else {
cout << pre << "-- " << a.etiqueta(n) << endl;
if (a.derecha(n)!=0 || a.izquierda(n)!=0) {// Si no es una hoja
pre += " |";
Esquema(a, a.derecha(n), pre);
pre.replace(pre.size()-4, 4, " ");
Esquema (a, a.izquierda(n), pre);
pre.erase(pre.size()-4, 4);
}
}
}

template
void Niveles(const ArbolBinario & a){
```

```
queue::Nodo> c;
typename ArbolBinario::Nodo n;

if (!a.empty()){
c.push(a.raiz());
while (!c.empty()){
n=c.front();
c.pop();
cout << a.etiqueta(n) << ' ';
if (a.izquierda(n) != 0)
c.push(a.izquierda(n));
if (a.derecha(n) != 0)
c.push(a.derecha(n));
}
}
cout << endl;
}

// Formato de entrada: n 1 n 2 n 4 x x n 5 x n 8 x x n 3 n 6 x x n 7 x x
// El ◆rbol: n 1 n 2 n 4 x x n 5 x n 8 x x n 3 n 6 x x n 7 x x
// tiene el esquema:
// -- 1
// |-- 3
// | |-- 7
// | -- 6
// -- 2
// |-- 5
// | |-- 8
// | -- x
// -- 4
//

int main(int argc, char *argv[]){
ArbolBinario a;
string pre;

cout << "Introduce el arbol en el formato:" << endl; cout << "n 1 n 2 n 4 x x n 5 x n 8 x x n 3 n 6 x x
n 7 x x" << endl; cin >> a;

cout << "El arbol: " << a << " tiene recorridos:" << endl;

cout << "Preorden: ";
preorden(a,a.raiz());
```

```
cout << endl;

cout << "Inorden: ";
inorden(a,a.raiz());
cout << endl;

cout << "Postorden: ";
postorden(a,a.raiz());
cout << endl;

cout << "y la altura es: " << Altura(a,a.raiz()) << endl;

cout << "Listado por niveles del arbol" << endl;
Niveles(a);

cout << "El arbol: " << a << " tiene el esquema:" << endl;
pre="";
Esquema(a,a.raiz(), pre);

refleja(a);

cout << "Listado por niveles del arbol reflejado" << endl;
Niveles(a);

cout << "El arbol reflejado tiene el esquema:" << endl;
pre="";
Esquema(a,a.raiz(), pre);

cout << "El arbol ejemplo es: " << endl; ArbolBinario final;
ArbolBinario b(8);
ArbolBinario c(5);
ArbolBinario d(2);
final.insertar_derecha(c.raiz(), b);
Esquema(final,final.raiz(),pre);

return 0;
}
```