Problema 1

public:

```
void push_back(const T& x) {
/* Pre: El paràmetre implícit és igual a la llista {e_1, ..., e_n}. */
/* Post: El paràmetre implícit és igual a la llista {e_1, ..., e_n, x}. */
  Node* aux = new Node;
  aux→info = x;
  aux→seg = nullptr;
  aux→ant = ultim_node;
  if (primer_node == nullptr) primer_node = aux;
  else ultim_node→seg = aux;
  ultim_node = aux;
  longitud++;
}
```

```
void interseccio_ordenada (const Llista & c2) {
/* Pre: El paràmetre implícit és igual a C1. C1 i c2 estan ordenades en ordre creixent
i no contenen elements repetits. */
/* Post: El paràmetre implícit conté els elements de C1 que pertanyen a la intersecció
de C1 i c2 en el mateix ordre en què estaven en C1. El punt d'interès del paràmetre
implícit apunta al seu inici. */
  Node* i1 = primer_node;
  Node* i2 = c2.primer_node;
  primer_node = nullptr;
  ultim_node = nullptr;
  act = nullptr;
  longitud = 0;
  while (i1 ≠ nullptr and i2 ≠ nullptr) {
    if (i1→info == i2→info) {
      if (primer_node == nullptr) {
        primer_node = i1;
        ultim_node = i1;
        act = i1;
      }
      else {
        ultim_node→seg = i1;
        i1→ant = ultim_node;
        ultim_node = i1;
      }
      i1 = i1→seg;
      i2 = i2→seg;
      longitud++;
    }
    else if (i1→info < i2→info) {
      Node* aux = i1;
      i1 = i1→seg;
      // Alliberem la memòria dels nodes de C1 que no pertanyen a la intersecció de C1 i c2.
      delete aux;
    }
    else i2 = i2→seg;
  }
```

```
    if (primer_node ≠ nullptr) {
        primer_node→ant = nullptr;
        ultim_node→seg = nullptr;
    }
    // Alliberem la memòria dels nodes de C1 que no pertanyen a la intersecció de C1 i c2.
    while (i1 ≠ nullptr) {
        Node* aux = i1;
        i1 = i1→seg;
        delete aux;
    }
}
```

Problema 2

public:

```
bool poda_subarbre(int x) {
/* Pre: El parámetro implícito es un árbol binario de enteros A. Los valores de
los nodos de A son todos distintos. */
/* Post: Si x es el valor de algún nodo de A, el resultado es cierto y el parámetro
implícito es el resultado de eliminar de A el nodo cuyo valor es x y todos sus
descendientes; en otro caso, el resultado es falso y el parámetro implícito no varía
(es decir, es A). */
  return poda_subjerarquia (primer_node, x);
}
```

private:

```
static bool poda_subjerarquia (Node_arbre*& m, int x) {
/* Pre: m = M. M apunta a NULL o a una jerarquía de nodos A. Los valores
   de los nodos de A son todos distintos. */
/* Post: Si x es el valor de algún nodo de A, el resultado es cierto, y m
   apunta a una jerarquía de nodos que es el resultado de 'eliminar' de A
   el nodo cuyo valor es x y todos sus descendientes; en otro caso, el
   resultado es falso y la jerarquía de nodos a la que apunta m no varía
   (es decir, es A). */
  bool encontrado;
  if (m ≠ nullptr) {
     if (m→info == x) {
        encontrado = true;
         esborra_node_arbre (m);
        m = nullptr;
     }
     else {
        encontrado = poda_subjerarquia (m→segE, x);
        if (not encontrado) encontrado = poda_subjerarquia (m→segD, x);
     }
  }
  else encontrado = false;
  return encontrado;
}

// Implementar esborra_node_arbre o una operación con otro nombre que haga
// lo mismo que esborra_node_arbre.
```