Si assuma di avere una classe C++ per l'implementazione di alberi n-ari in sola lettura e una per l'implementazione di una lista, che presentino le seguenti funzioni:

```
template< class T >
class ReadOnlyTree {
 public:
        typedef int Nodo;

        ReadOnlyTree();
        bool vuoto() const;
        Nodo radice() const;
        Nodo padre(Nodo) const;
        Nodo primofiglio(Nodo) const;
        Nodo succfratello(Nodo) const;
        bool ultimofratello(Nodo) const;
        bool foglia(Nodo) const;
        leggi(Nodo) const;
        void scrivi(Nodo, const T &) const;
}
```

```
template< class T >
class MyList {
 public:
        typedef Cella* position;

        MyList();
        bool vuota() const;
        T leggi(position) const;
        void scrivi(const T &, position);
        position begin() const;
        position successivo(position) const;
        position precedente(position) const;
        void inserisci(const T &, position);
        void rimuovi(position);
        bool ultimo(position);
}
```

Si assuma inoltre di avere a disposizione due funzioni get_albero_1 ( ) e get_albero_2 ( ), ciascuna delle quali restituisce un albero n-ario.

Scrivere la funzione *ammette_inverso*(…) che prenda in input due alberi n-ari albero1 e albero2 interi e restituisca una lista di interi contenente i nodi di albero1 avente l'inversa nei nodi dell'albero2.

Esempio: Per poter verificare l'inverso di un numero, bisogna moltiplicarlo per il suo reciproco e se il risultato è 1, allora il numero ammette l'inversa.

Scrivere dunque il main che chiami le due funzioni get_albero_1 ( ) e get_albero_2 ( ) per ottenere due alberi, li passi alla funzione *ammette_inverso* (…) e ne stampi la lista restituita.