



# Elias De Hondt

P e r f e c t i o n   I s   E v e r y t h i n g



---

---

---





1. What are the differences between array-backed lists (`std::vector`) and linked lists (`std::forward_list` and `std::list`)? When would you use an array-backed list and when would you use a linked list?
  - **Vector:** gebruikt aaneengesloten geheugen, biedt snelle random access, maar invoegen/verwijderen in het midden is duur.
  - **List/forward\_list:** bestaan uit losse nodes met pointers, maken invoegen/verwijderen efficiënt, maar random access is traag en geheugenverbruik hoger.
  - **Gebruik:** vector bij veel lezen en indexeren, list bij veel invoegingen/verwijderingen in het midden.
2. Why do we need to be able to seed our random number generator?
  - Zonder seed krijg je altijd dezelfde reeks getallen; met seed (bv. huidige tijd) krijg je variatie en echte willekeur.
3. The documentation for `std::vector::push_back` mentions that under certain conditions, all iterators and references are invalidated. What does this mean?
  - Wanneer de vector moet heralloceren (groeien), verschuift het onderliggende geheugen. Alle oude iterators en verwijzingen naar elementen zijn dan ongeldig.
4. We notice that almost every algorithm in the `<algorithm>` header takes a begin and end iterator as an argument. Why do you think the writers of the C++ standard chose to do this?
  - Dit maakt algoritmes generiek: ze werken met elk container-type en elk bereik, niet alleen met specifieke datastructuren.