Corso di Laboratorio di Programmazione

A.A. 2024/25

# Laboratorio Extra
# Smart pointer

## Esercizio 1

Implementare un semplice `unique_ptr` che supporta solo:

- costruttore
- distruttore
- `->`
- `*`
- `release()`

Non implementare altre funzionalità (es., assegnamento, costruttore di copia, ecc.).

## Esercizio 2

Durante il modulo sugli smart pointer abbiamo visto che una debolezza che non può essere gestita dal compilatore è l'inizializzazione di due `unique_ptr` con lo stesso puntatore. Per risolvere questo problema è possibile usare `make_unique()`, che restituisce `unique_ptr` senza rendere accessibile il relativo puntatore. `make_unique()` è una funzione template in cui il template specificato corrisponde al tipo puntato dallo `unique_ptr`, mentre gli argomenti sono utilizzati per inizializzare l'oggetto puntato. Leggere la documentazione:

https://docs.w3cub.com/cpp/memory/unique_ptr/make_unique

(con particolare attenzione alla sezione "example" e a come sono creati `unique_ptr` a oggetti di classe `Vec3`).

Successivamente:

- creare con `make_unique()` un `unique_ptr` a uno `std::vector<int>` di 10 elementi (la dimensione deve essere passata in fase di costruzione);
- inizializzare tutti gli elementi con un ciclo;
- aggiungere altri 4 elementi con `push_back()`;
- stampare tutti i valori del vettore.