UNIVERSITÉ DU LUXEMBOURG

Smart Pointers

Parallel Computing

Goals

- ✯ Design a smart pointer in C++.
- ✯ Think in groups of 2 or 3 students.

Exercise 1 – Smart pointer

The problem of memory allocation in C and C++ is that it must be managed by the user. In particular, when you return a pointer from a function, e.g.:

```
int* make_array(size_t n);
```

who has the responsibility to delete the returned object? The user or another `destroy_array` from the API? What this code does not highlight is who has the *ownership* of the pointer. It can lead to bugs such as double-free (calling free two times), using a pointer after it has been free (leading to segfault), memory leak (never freeing the allocated memory).

To make ownership clear, the idea is to encapsulate a pointer inside a class. When the object goes out-of-scope, the destructor is called and the pointer can be deleted. Try to design a smart pointer with the following things in mind:

- What happens when the object is copied?
- What happens when the object is moved?
- What happens when the object is destroyed?

Is it possible to come up with a design that allows several structure to share a pointer? And the last object alive will automatically delete the pointer?