

Move Semantics and Smart Pointers Solutions

- How can the copy operators of a class be disabled?
 - Put =delete after their prototype (C++11 onwards)
 - Make them private (all versions)
- What happens when the copy operators of a class are disabled?
 - Instances of the class cannot be copied (including pass and return by value)

- Disable the copy operators in your class from the previous video. What effect does this have?
 - Compiler error - use of deleted function

- What is unusual about `unique_ptr`?
 - It cannot be copied, but can still be passed to and returned from functions
- Describe how the implementation of `unique_ptr` causes these unusual features
 - The copy operators are deleted
 - When passing to and returning from functions, the move operators are used
 - Ownership of the underlying data is passed into the function

- Write a function which takes a `unique_ptr` instance by value
- Write a program to exercise this function
- Add a function to your program which returns a `unique_ptr` instance by value
- Alter your program to exercise this function