Practical 3 (due 2022-08-12 @ 09:00)

The purpose of this practical is for you to become familiar with the basics of composition and delegation.

You may use the memo for Practical 2 in the creation of Practical 3. This will not be considered as plagiarism <u>if you include a comment indicating you have done so</u>.

You will need to create a class called `CanvasCapper`

- This class includes a public void member function called `applyRangeRules()`
  - The member function must take a `Canvas2D` reference as input as well as two real values for `max` and `min`
  - The member function must loop through all the pixels in the pixel array and make sure that the value of each pixel is between the range specified by the `max` and `min` parameters. The member function must also use ensure that the provided `max` and `min` parameters themselves are in the range [0, 255]. Any values which are out of range must be capped.
- The `Canvas2D` must be updated so that
  - it **has-a** `CanvasCapper` whose life-cycle it manages directly (the `CanvasCapper` must be instantiated from the free-store when the `Canvas2D` is created and de-allocated when the `Canvas2D` is destroyed).
  - it has an `applyRangeRules()` member function which it delegates to its contained `CanvasCapper`
  - Note that the `Canvas2D` needs to know about the `CanvasCapper` and the reverse is true. This can cause problems with compilation that need to be solved using forward declarations. Please see the following tutorial for an example of that problem being solved: https://youtu.be/WTQP0JQ7tBY.

Mark sheet		
	Design	10
	CanvasCapper class with applyRangeRules member function	10
	Containment relationship	10
	Allocation of contained class	10
	De-allocation of contained class	10
	Delegation of applyRangeRules	10
	Demonstration of functionality in a main function	10
	Total	/70