University of Toronto, Department of Electrical and Computer Engineering

ECE 1724-F1 — Performant Software Systems with Rust

Quiz 4 — Sample Quiz 3 — Answer Key

Wednesday, November 11, 2026

1. Ⓒ **A concrete implementation defines methods only for that type.**

The field annotation makes `writer` a `CappedWriter<u16>`. The method belongs only to `CappedWriter<u32>`; Rust does not convert the receiver to make the call work. Concrete implementation blocks for generic structs are allowed. (Book 10.1.) Teaching variation of Rustrace's CappedWriter, adding a concrete-type-only method.

2. Ⓐ **A bounded implementation also works for nested wrappers.**

`Log` implements `EditorEffects`, so its wrapper does too. The outer wrapper therefore also satisfies the bound. Each wrapper adds one: `3 + 1 + 1 = 5`. (Book 10.2.) Adapted from Rustrace's SyntaxEffects forwarding implementation.

3. Ⓓ **Return-position impl Trait hides one concrete type.**

Both branches must return the same concrete type. Both structs implementing `EditorEffects` is insufficient, and passing `true` does not exempt the other branch from type checking. (Book 10.2.) Original teaching variation using the Rustrace EditorEffects pattern.

4. Ⓑ **The associated type fixes the next method's result type.**

Here `Self::Item` is `u32`, so `next` returns `Option<u32>`. The calls increment the same iterator to five and then six; debug formatting retains the `Some` wrappers. (Book 20.2, associated types.) Original miniature iterator inspired by Rustrace's Iterator<Item = char> contracts.

5. Ⓐ **A supertrait requirement is not an automatic subtrait implementation.**

Implementing `EditorEffects` only satisfies the prerequisite. Add `impl WorkspaceEffects for Log {}` to opt into the subtrait and its default method; then the program prints `true`. (Book 20.2, supertraits.) Simplified from Rustrace's WorkspaceEffects relationship.

Reading: *The Rust Programming Language*, 10.1: Generic Data Types, 10.2: Traits, and 20.2: Associated Types and Supertraits only.