Generic Lifetimes

Learn to Code with Rust

A Review of Generics

- A **generic** is a placeholder for a future type.
- Generics add flexibility by not hardcoding an exact type.
- Code can use a variety of types in place of the generic.

Generic Lifetimes vs. Concrete lifetimes

- A **concrete lifetime** is the region of code that a value exists in the program (the time it lives in its memory address).
- A **generic lifetime** is more abstract. It is a hypothetical lifetime, a non-specific lifetime, a future lifetime that can vary.
- We can annotate generic lifetimes in our code. This enables functions that are flexible enough to handle varying lifetimes.

Lifetime Annotations

- A **lifetime annotation** is a name or label for a lifetime.
- Lifetime annotations don't change the reference's lifetime. They don't affect the logic in any way.
- A lifetime annotation is a piece of metadata that we provide to the borrow checker so that it can validate that references are valid.