The `impl Trait` return type captures lifetime parameters that do not appear within the `impl Trait` itself.

Erroneous code example:

```
use std::cell::Cell;

trait Trait<'a> { }

impl<'a, 'b> Trait<'b> for Cell<&'a u32> { }

fn foo<'x, 'y>(x: Cell<&'x u32>) -> impl Trait<'y>
where 'x: 'y
{
    x
}
```

Here, the function `foo` returns a value of type `Cell<&'x u32>`, which references the lifetime `'x`. However, the return type is declared as `impl Trait<'y>` – this indicates that `foo` returns “some type that implements `Trait<'y>`”, but it also indicates that the return type **only captures data referencing the lifetime `'y`**. In this case, though, we are referencing data with lifetime `'x`, so this function is in error.

To fix this, you must reference the lifetime `'x` from the return type. For example, changing the return type to `impl Trait<'y> + 'x` would work:

```
use std::cell::Cell;

trait Trait<'a> { }

impl<'a,'b> Trait<'b> for Cell<&'a u32> { }

fn foo<'x, 'y>(x: Cell<&'x u32>) -> impl Trait<'y> + 'x
where 'x: 'y
{
    x
}
```