Please use the following message instead of the error message on page 161.

error[E0277]: the `?` operator can only be used in a function that returns

`Result` or `Option` (or another type that implements `std::ops::Try`)

--> src/main.rs:4:13

|

4 | let f = File::open("hello.txt")?;

| ^^^^^^^^^^^^^^^^^^^^^^^^ cannot use the `?` operator in a

function that returns `()`

|

= help: the trait `std::ops::Try` is not implemented for `()`

= note: required by `std::ops::Try::from\_error`

Then, please replace the highlighted paragraph on page 161 with this text

This error points out that we’re only allowed to use the ? operator in a function that returns Result<T, E>. When you’re writing code in a function that doesn’t return Result<T, E>, and you want to use ? when you call other functions that return Result<T, E>, you have two choices to fix this problem. One technique is to change the return type of your function to be Result<T, E> if you have no restrictions preventing that. The other technique is to use a match or one of the Result<T, E> methods to handle the Result<T, E> in whatever way is appropriate.

The main function is special, and there are restrictions on what its return type must be. One valid return type for main is (), and conveniently, another valid return type is Result<T, E>, as shown here:

use std::error::Error;

use std::fs::File;

fn main() -> Result<(), Box<dyn Error>> {

let f = File::open("hello.txt")?;

Ok(())

}

The Box<dyn Error> type is called a trait object, which we’ll talk about in the section “Using Trait Objects that Allow for Values of Different Types” in Chapter 17. For now, you can read Box<dyn Error> to mean “any kind of error.” Using ? in a main function with this return type is allowed.

prod: check xref