[[questions]]
type = "Tracing"
prompt.program = """
pub mod rectangle {
    #[derive(Copy, Drop)]
    pub struct Rectangle {
        width: u64,
        height: u64
    }
}
fn main() {
    let r = rectangle::Rectangle { width: 10, height: 20 };
    println!("{}", r.width);
}
"""
answer.doesCompile = false
context = """
While the `Rectangle` structure is public, the fields `width` and `height` are not marked as `pub`.
Therefore instantiating these fields outside of the `rectangle` module is not allowed. It's also impossible to access the field `width` to print it.
This program would compile if the structure were changed to:
```
pub struct Rectangle {
    pub width: u64,
    pub height: u64
}
```
"""
id = "ba0f75cb-dfa6-461f-906e-e9eeaea192e6"
[[questions]]
type = "Tracing"
prompt.program = """
pub mod a {
    pub mod b {
        pub fn f() {
            println!("b1");
        }
        pub mod c {
            pub fn f() {
                println!("c1");
            }
        }
    }
    pub fn entry() {
        super::b::c::f();

```
    }
}
pub mod b {
    pub fn f() {
        println!("b2");
    }
    pub mod c {
        pub fn f() {
            println!("c2");
        }
    }
}
fn main() {
    a::entry();
}
"""
answer.doesCompile = true
answer.stdout = "c2"
context = """
`entry` uses the path `super::b::c::f`. `entry` is within the module `a`, so `super` refers to the parent module of `a`, which is the root crate.
Then the child `b` of the root is the outermost module `b`, whose child `c` contains a function `f` that prints "c2".
"""
id = "2f0361e2-b5d4-46cd-ac72-dca753a553c8"
```