An attempt was made to retrieve an associated type, but the type was ambiguous.

Erroneous code example:

```
trait T1 {}
trait T2 {}

trait Foo {
    type A: T1;
}

trait Bar : Foo {
    type A: T2;
    fn do_something() {
        let _: Self::A;
    }
}
```

In this example, `Foo` defines an associated type `A` . `Bar` inherits that type from `Foo` , and defines another associated type of the same name. As a result, when we attempt to use `Self::A` , it's ambiguous whether we mean the `A` defined by `Foo` or the one defined by `Bar` .

There are two options to work around this issue. The first is simply to rename one of the types. Alternatively, one can specify the intended type using the following syntax:

```
trait T1 {}
trait T2 {}

trait Foo {
    type A: T1;
}

trait Bar : Foo {
    type A: T2;
    fn do_something() {
        let _: <Self as Bar>::A;
    }
}
```