Something other than a type or an associated type was given.

Erroneous code example:

```
enum Rick { Morty }

let _: <u8 as Rick>::Morty; // error!

trait Age {
    type Empire;
    fn Mythology() {}
}

impl Age for u8 {
    type Empire = u16;
}

let _: <u8 as Age>::Mythology; // error!
```

In both cases, we're declaring a variable (called `_`) and we're giving it a type. However, `<u8 as Rick>::Morty` and `<u8 as Age>::Mythology` aren't types, therefore the compiler throws an error.

`<u8 as Rick>::Morty` is an enum variant, you cannot use a variant as a type, you have to use the enum directly:

```
enum Rick { Morty }

let _: Rick; // ok!
```

`<u8 as Age>::Mythology` is a trait method, which is definitely not a type. However, the `Age` trait provides an associated type `Empire` which can be used as a type:

```
trait Age {
    type Empire;
    fn Mythology() {}
}

impl Age for u8 {
    type Empire = u16;
}

let _: <u8 as Age>::Empire; // ok!
```