The `self` keyword was used inside of an associated function without a "`self` receiver" parameter.

Erroneous code example:

```
struct Foo;

impl Foo {
    // `bar` is a method, because it has a receiver parameter.
    fn bar(&self) {}

    // `foo` is not a method, because it has no receiver parameter.
    fn foo() {
        self.bar(); // error: `self` value is a keyword only available in
                    //        methods with a `self` parameter
    }
}
```

The `self` keyword can only be used inside methods, which are associated functions (functions defined inside of a `trait` or `impl` block) that have a `self` receiver as its first parameter, like `self`, `&self`, `&mut self` or `self: &mut Pin<Self>` (this last one is an example of an "arbitrary `self` type").

Check if the associated function's parameter list should have contained a `self` receiver for it to be a method, and add it if so. Example:

```
struct Foo;

impl Foo {
    fn bar(&self) {}

    fn foo(self) { // `foo` is now a method.
        self.bar(); // ok!
    }
}
```