Review

You learned a lot this lesson: congrats on finishing! Here's what you've covered:

Call a method with its name and parentheses:

```
VisitPlanets();
```

Store a method's returned value in a variable:

```
double result = Math.Round(3.14159, 2);
```

Define a basic method with the following syntax:

```
static void VisitPlanets()
{
}
```

Every time an application is started, the `Main()` method is called.

Values passed to a method are called *arguments*. When defined in the method, they are *parameters*.

Method parameters can only be used within the method body.

Method parameters can be *optional* if given a default value using equals `=` syntax:

```
static void VisitPlanets(int numberOfPlanets = 0)
```

When calling a method, pass arguments by position or by name. If using names, use the colon (`:`) syntax:

```
VisitPlanets(numberOfPlanets: 9);
```

In *method overloading*, multiple methods can have the same name, as long as they have different method *signatures*.

A method *signature*is a method's name and parameter types in order.

☑Instructions

1.

Make sure you know how to apply all of these concepts before moving on!

To pass this last exercise:

- Call `NamePets()` with two arguments
- Call `VisitPlanets()`, and specify only the `numberOfPlanets`

Hint

Call both methods in `Main()`.

Here's how to call a method with a named argument.

```
YourMethodName(d: 2);
```