Named Arguments

Say your method has lots of optional parameters, but you only want to specify one when you call it.

In this example, your method has five optional parameters:

```
static void YourMethodName(int a = 0, int b = 0, int c = 0, int d = 0, int e
= 0) {...}
```

When you call the method, you only want to specify `d`. But calling the method this way would set `a` to `4`, not `d`!

```
YourMethodName(4);
```

Refer to the parameter by its name instead:

```
YourMethodName(d: 4);
```

With named arguments, you can list them in any order:

```
YourMethodName(d: 4, b: 1, a: 2);
```

You can also mix named arguments with positional arguments, but positional arguments MUST come before named arguments:

```
YourMethodName(2, 1, d: 4) // a is 2, b is 1, d is 4
YourMethodName(d: 4, 2, 1) // Error!
```

Named arguments aren't always necessary, but they can be useful when:

- a method has many optional parameters
- you want to differentiate between similar arguments

Instructions

1.

The `VisitPlanets()` method has some new optional parameters.

First, call the method in `Main()` with no arguments.

2.

Call the method again, but define only the `numberOfPlanets` parameter as `2`.

Hint

Refer to the parameter by its name:

```
YourMethodName(d: 4);
```

3.

Call the method one more time, now defining the `numberOfPlanets` as `2` and `name` with your name.

Hint

Refer to the parameter by its name:

```
YourMethodName(a: "foo", d: 4);
```