QUIZ

Why are destructors helpful in C++?

A destructor allows you to conduct any final cleanup necessary before an object is destroyed.

This helps you prevent memory leaks in your program.

Without defining a destructor, there is no way to destroy an object.

A destructor allows you to destroy a class.

Build an empty class for cities.

```
class City {

  // class members go here

}    ;
```

You got it!

What is not possible to do with a `Country` object?

```
class Country {
private:

  std::string name;

public:

  Country(std::string new_name)
    : name(new_name) {}

  std::string get_name() {

    return name

  }

};
```

You cannot change a `Country` instance's `name` attribute.

Correct. You can only retrieve the `name` attribute.

Instantiate a `City` object called `seoul` with (in this order):

- `name` set to `"Seoul"`.
- `population` set to `9776000`.

```
City seoul ( "Seoul" , 9776000 );
```

👏 You got it!

Which of the following are class members of `Country`?

```
class Country {
private

  std::string name;

public:

  Country(std::string new_name)
    : name(new_name) {}

  std::string get_name() {

    return name

  }

};
```

`name`, the `Country` constructor, and `.get_name()` are all class members.

👏 Any data attributes and methods contained in a class are considered class members.

Define a method `.add_resident()` for a `City` class.

(This method is defined in a **.cpp** file, not inside the class.)

`City` has a `population` attribute that tracks the number of residents.

```
#include "city.hpp"

void City::add_resident () {

  population++;

}
```

👏 You got it!