

std::bind() Solutions

Generalized predicate

- Imagine we want to call `count_if()` and use the following function as a predicate

```
bool match(const string& animal, const string& value) {  
    return animal == value;  
}
```

- Why does this not work?
 - `count_if`'s predicate can only take one argument (the element)
- Suggest two different ways it could be made to work as intended
 - `std::bind`
 - Lambda with capture and lambda-local variable

std::bind()

- Explain what std::bind() does
 - It performs a partial function call
 - std::bind() returns a callable object in which some arguments have been captured, while other arguments are left for the completion of the function call
 - "Normal" arguments (literals or variables) are captured
 - "Placeholder" arguments will be supplied when the returned object is invoked and forwarded to the original function

Placeholder arguments

- What syntax is used for placeholder arguments?
 - The first argument to the callable object is `_1`
 - The second argument to the callable object is `_2`
 - And so on
 - These are defined in the `std::placeholders` namespace

std::bind() example

- Rewrite the example from the first slide so that it works correctly, using std::bind()

Lambda equivalent

- Rewrite the example from the first slide so that it works correctly, using a lambda with capture and a lambda-local variable