- Solution

Let's have a look at a solution review for the last exercise.

WE'LL COVER THE FOLLOWING

Solution Review

```
// templateFactorial.cpp

#include <iostream>

template <int N>
struct Factorial{
    static int const value = N * Factorial<N-1>::value;
};

template <>
struct Factorial<1>{
    static int const value = 1;
};

int main(){
    std::cout << Factorial<10>::value << std::endl;
}
```

Explanation

The `Factorial` function calls itself recursively by storing the result of each iteration in `value`. For the base case, the result is multiplied by 1 in line 12 and the answer is stored in value.

> To be precise, what looks like a recursion, like in the case of `Factorial`, is

not a recursion. Each invocation of `Factorial<N>` triggers a new instantiation `Factorial<N-1>` in line 7.

In the next lesson, we'll discuss type traits.