- Solution

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

WE'LL COVER THE FOLLOWING

- Solution Review
  - Explanation

Solution Review

```cpp
// templateInstantiationInvalid.cpp

#include <iostream>
#include <vector>

template <int Nom, int Denom>
class Rational{
public:
    int getFloor(){
        return Nom / Denom;
    }
};

// template int Rational<5, 0>::getFloor();

int main(){

  std::cout << std::endl;

  Rational<5, 3> rat1;
  std::cout << "rat1.getFloor(): " << rat1.getFloor() << std::endl;


  Rational<5, 0> rat2;
  // std::cout << "rat2.getFloor(): " << rat2.getFloor() << std::endl;

  std::cout << std::endl;

}
```

Explanation

In the above code, we have called the `getFloor` function for 5 and 3 in line 9, and it returns 1. To invoke the function for an invalid call, we can give the arguments 5 and 0 which gives an error.

In the next lesson, we’ll discuss variadic templates.