- Solution

This lesson explains the solution for the exercise in the previous lesson.

WE'LL COVER THE FOLLOWING

- Explanation

The solution to the previous exercise can be found below:

```
#include <iostream>
#include <typeinfo>

template<typename T1, typename T2>
auto add(T1 first, T2 second) -> decltype(first + second){
    return first + second;
}

int main(){

  std::cout << std::endl;

  // -> int
  std::cout << typeid( add(1, false) ).name() << std::endl;
  std::cout << typeid( add('a', 1) ).name() << std::endl;
  std::cout << typeid( add(false, false) ).name() << std::endl;

  // -> double
  std::cout << typeid( add(true, 3.14) ).name() << std::endl;
  std::cout << typeid( add(1, 4.0) ).name() << std::endl;

  std::cout << std::endl;

}
```

Explanation

- From lines 14 to 16, we can see various arithmetic operations that get implicitly cast to the `int` data type.

- When booleans and integers are operated together, the result is a cast to `int`.
- From lines 19 to 20, we can see arithmetic operations that get implicitly cast to the `double` data type.
- The result of arithmetic operations between integers and doubles is a double.

Further information: typeid

This concludes the concept of automatic type deduction. Next, we'll tackle explicit casts in C++.