

Trailing Return

Chapter 1 Safe Features

```
auto getOperation(Operation kind) // (1) function taking a kind of Operation
-> double (Calculator::*)(double);
    // (2) returning a pointer to a Calculator member function taking a
    //     double and returning a double
```

Using this divide-and-conquer approach, writing such functions becomes fairly straightforward. Declaring a **higher-order function** that takes a function pointer as an argument might be even easier to read if a type alias is used via **typedef** or, as of C++11, **using**.

See Also

- “**decltype**” (§1.1, p. 25) describes how function declarations might use **decltype** either in conjunction with, or as an alternative to, trailing return types.
- “**auto Return**” (§3.2, p. 1182) explains that leaving the return type to deduction shares syntactical similarities with trailing return types but brings with it significant pitfalls when migrating from C++11 to C++14.

Further Reading

- Those interested in consistency of style — a la east **const** versus **const** west — can find such a discussion involving *trailing* versus *classic* return types in **mertz18**.