Exercise 9

Author: Elena Pfefferlé

General:

In this code we have 3 classes `class A`, `class B` and `class C`. `class B` inherits from `class A` but `class C` inherits from `class B`.

We also have a function `void func()` which take as argument a reference to an object of `class A`.

Line 1-3 - A a; B b; C c;

Here we create an object `a` of `class A`, an object `b` of `class B` and an object `c` of `class C`. This has no output.

- **guessed output** : `nothing`

Line 4-5 - a.f(); a.g();

Here we call the function `f();` and `g();` of `class A`.

- **guessed output** : "A::f A::g "

Line 6-7 - b.f(); b.g();

Here we call the functions `f()` and `g()` of `class B`.

- **guessed output** : "B::f B::g "

Line 8-9 - c.f(); c.g();

Here we call the functions `f()` and `g()` of `class C`.

- **guessed output** : "C::f C::g "

Line 10 - func(a);

We pass the object `a` into `func()`, we get the same output as with lines 4-5.

- **guessed output** : "A::f A::g "

Line 11 - func(b);

We pass the object `b` into `func()`, however, since `a.g()` is not a virtual function (not overriden by derived classes) and `b.g()` is not `const` (therefore not overriding base member function); `b.g()` will output `a.g`.

- **guessed output** : "B::f A::g "

Line 12 - func(c);

We pass the object `c` , but `c.f()` is not `const` therefore we will see the output of `b.f()` instead. Furthermore, since `a.g()` is not is not `virtual` , we will not see the output from `c.g()` .

- **guessed output** : "B::f A::g "

Guessed output

```
A::f A::g B::f B::g C::f C::g A::f A::g B::f A::g B::f A::g
```