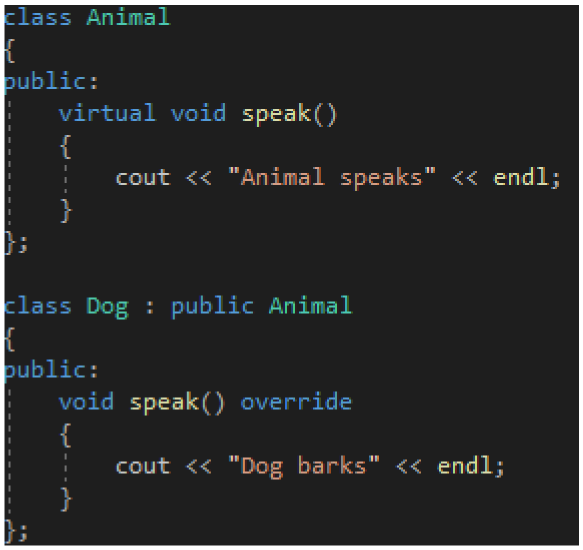
Object Oriented programming and software engineering – Lab 9

Adam Korytowski – 2025

1. Virtual functions

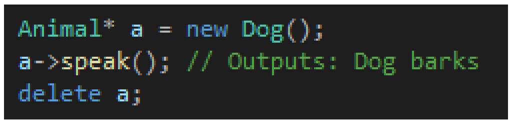
In C++, virtual functions are member functions in a base class that you can override in derived classes. They allow for runtime polymorphism, meaning the function that gets called is determined by the type of the object pointed to, not the type of the pointer.

When you declare a function as virtual in a base class, it tells the compiler: “If this function is overridden in a derived class, and you're calling it through a pointer or reference to the base class, then use the derived class’s version.”



```
class Animal
{
public:
    virtual void speak()
    {
        cout << "Animal speaks" << endl;
    }
};

class Dog : public Animal
{
public:
    void speak() override
    {
        cout << "Dog barks" << endl;
    }
};
```



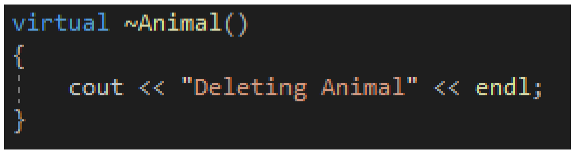
```
Animal* a = new Dog();
a->speak(); // Outputs: Dog barks
delete a;
```

Without the virtual keyword, this would output "Animal speaks", because ‘*a*’ is of type Animal*.

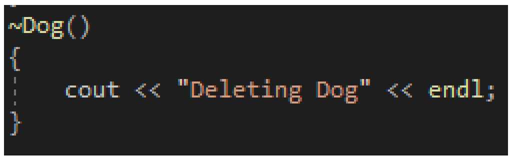
2. Virtual destructor

The point of a virtual destructor is to ensure proper cleanup of resources when you're deleting an object through a base class pointer.

If you delete a derived class object using a base class pointer and the destructor is not virtual, the derived class's destructor won't get called, which can lead to memory leaks or incomplete cleanup.



```
virtual ~Animal()
{
    cout << "Deleting Animal" << endl;
}
```



```
~Dog()
{
    cout << "Deleting Dog" << endl;
}
```

If your class has virtual functions, you should always make the destructor virtual too.

3. Tasks (3 pts each)

- implement one virtual method for two of your base classes, then override them in two child classes for each class (4 total overriding methods)
- in one of your classes implement virtual destructor, in other class non-virtual destructor. Present the difference, showing that in the first class both destructors are called (from base and child class), in the second one only base class's destructor is called.