Some “words of wisdom” from Michael about classes:

* Simulate some animals in an environment, living their lives.
* Some herbivores (plant eaters) and carnivores (meat eaters) and some plants with fruit.
  + Wolf, Bear (meat eater)
  + Rabbit, Salmon (fish) (plant eater)
  + Carrots (fruit)
* These animals wander around, looking for food, and that’s it for now.

Classes:

* Make a header file with JUST the class definition.
  + Make a “header guard” (#ifndef, #define, #endif or #pragma once)
  + DON’T include any code unless it’s “templated”
  + Usually the .h file is the same name as the class, but *it doesn’t have to be*.
* All the “code” is in the “implementation” (i.e. one or more CPP files):
  + You place a “scope” in front of the “methods”
  + If it’s class cAnimal, then all the functions have cAnimal:: in front of them
* Default values:
  + In pre 2003 C++, you could only initialize integers, because it’s at compile time.
    - class cAnimal:
      * int size = 3;
    - any other type, it’s an error
  + The conventional way to initialize default values in in the constructor.
  + If you had many constructors, you might make a specific “init” method or something.

In C++, there are many types of inheritance:

* Non-virtual or virtual:
  + Just code reuse and overriding
  + Is it “polymorphism”
* Public, private, protected, and friend
* Without the keyword “virtual” in ANY place in the class:
  + It will be set at COMPILE time to decide what functions are called.
  + You can override this by casting it as the “derived” class or whatever

NON-polymorphic: functions aren’t using “virtual”:

* Function is determined at compile time.
* Based on the type of object it is, NOT what you created:
  + cAnimal\* pX = new cRabbit(); 🡪 is an cAnimal
  + cRabbit\* pX = new cRabbit(); 🡪 is a Rabbit

Polymorphic functions: i.e. there’s a “virtual” in front of the method:

* Compiles the class with a v-table.
* At RUN-TIME will determine what method to call
* Based on the what type you CREATE (instantiate) NOT the type the variable is
  + cAnimal\* pX = new cRabbit(); 🡪 is an cRabbit
  + cRabbit\* pX = new cRabbit(); 🡪 is also a Rabbit

Warning:

* It’s strange to mix these up, like to use “virtual” in only SOME of the methods.
* Usually, you make ALL of them virtual.
* BUT you have to do this individually.
* You need to have a “virtual destructor” on all the classes in the inheritance chain
  + Or you’ll get a memory leak