- Example

In this lesson, we'll look at the example of constructor inheriting with the help of the keyword `using`.

Example: Constructor inheriting

```
#include <iostream>
#include <string>

class Base{
 public:

    Base() = default;
    Base(int i){
      std::cout << "Base::Base("<< i << ")" << std::endl;
    }

    Base(std::string s){
      std::cout << "Base::Base("<< s << ")" << std::endl;
    }
};

class Derived: public Base{
  public:

    using Base::Base;

    Derived(double d){
      std::cout << "Derived::Derived("<< d << ")" << std::endl;
    }

};

int main(){

  // inheriting Base
  Derived(2011);          // Base::Base(2011)

  // inheriting Base      // Base::Base(C++0x)
  Derived("C++0x");
  // using Derived
  Derived(0.33);          // Derived::Derived(0.33)



}
```

Explanation

In this example, we have created two classes, i.e., `Base` and `Derived`. The `Derived` class inherits the `Base` class publicly. When we call the `Derived` class object with the keyword `using`, it calls the relative constructors. This can be clearly seen in the objects created in the `main` section. For integers and strings, it called the `Base` class constructors and for doubles, it calls the `Derived` class constructor.

In the next lesson, we'll solve an exercise to get a better grip on constructor inheriting.