Quiz 5 discussion

Victor Eijkhout, Susan Lindsey

Fall 2025

last formatted: October 29, 2025

Exercise 1

Given two classes with similar functionality:

```cpp
// object/company1.cpp
class Worker {
private:
  int earned{0};
public:
  void work_for( int months ) {
    earned += months * 2000; };
  int wages() { return earned; };
};
```

```cpp
// object/company1.cpp
class Boss {
private:
  int earned{0};
  int workers{0};
public:
  Boss( int w )
    : workers(w) {};
  void work_for( int months ) {
    earned += months *
      ( 3000 + 500 * workers ); };
  int wages() { return earned; };
};
```

```cpp
// object/company1.cpp
Worker alice;  alice.work_for(2);
Worker bob;    bob.work_for(2);
Boss carol(2); carol.work_for(2);
println( "{}; {}, {}",
         carol.wages(),alice.wages
     (),bob.wages()
         );
```

1. Boss method overrides

Override the *work_for* method in the *Boss* class:

```cpp
// object/company2.cpp
class Worker {
protected:
  int earned{0};
public:
  virtual void work_for( int months
      ) {
    earned += months * 2000; };
  int wages() { return earned; };
};
```

```cpp
// object/company2.cpp
class Boss : public Worker {
private:
  int workers{0};
public:
  Boss( int w )
    : workers(w) {};
  void work_for( int months )
      override {
    earned += months *
      ( 3000 + 500 * workers ); };
};
```

(No need to create a common base class for *Boss* and *Worker* though that is not wrong.)

2. Boss salary

Define `Boss` as worker with different salary:

```cpp
// object/company3.cpp
class Worker {
protected:
  int earned{0},monthly_salary{0};
public:
  Worker( int rate )
    : monthly_salary(rate) {};
  virtual void work_for( int months ) {
    earned += months * monthly_salary; };
  int wages() { return earned; };
};
```

```cpp
// object/company3.cpp
class Boss : public Worker {
public:
  Boss( int w,int rate )
      : Worker( rate + 500 * w ) {};
};
```

- This changes the Application Programmer Interface (API)
- Can not account for changing numbers of workers.

3. Use base class method

Express that part of the `Boss` salary is from his role as `Worker`:

```cpp
// object/company4.cpp
class Worker {
protected:
  int earned{0},monthly_salary {2000};
public:
  virtual void work_for( int months ) {
    earned += months * monthly_salary; };
  int wages() { return earned; };
};
```

```cpp
// object/company4.cpp
class Boss : public Worker {
private:
  int workers{0};
public:
  Boss( int w )
    : workers(w) { monthly_salary = 3000; };
  void work_for( int months ) override {
    Worker::work_for(months);
    earned += months * 500 * workers; };
};
```

4. Virtual base class

Base class has method defined as `= 0`:

```cpp
// object/company5.cpp
class Job {
protected:
  int earned{0},monthly{0};
public:
  virtual void work_for( int months ) = 0;
  int wages() { return earned; };
};
class Worker : public Job {
public:
  Worker() { monthly = 2000; };
  virtual void work_for( int m ) override {
    earned += m * monthly; };
};
```

```cpp
// object/company5.cpp
class Boss : public Job {
private:
  int workers{0};
public:
  Boss( int w )
    : workers(w) { monthly = 3000; };
  virtual void work_for( int m ) override {
    earned += m * ( 3000 + 500 * workers ); };
};
```