CSC 254
Quiz 3 — Object Orientation
November 25th, 2003

1. Some language designers argue that object orientation eliminates the need for nested subroutines. Do you agree? Why or why not?

2. In several object-oriented languages, including C++ and Eiffel, a derived class can hide members of the base class. In C++, for example, we can declare a base class to be `public`, `protected`, or `private`:

```
class B : public A {
   // public members of A are public members of B
   // protected members of A are protected members of B
...
class C : protected A {
   // public and protected members of A are protected members of C
...
class D : private A {
   // public and protected members of A are private members of C
...
```

   IN all cases, `private` members of `A` are inaccessible to methods of `B`, and `C`.

   Consider the impact of `protected` and `private` base classes on dynamic method binding. Under what circumstances can a reference to an object of class B, C, or D be assigned into a variable of type `A*`?

3. What happens to the implementation of a class if we redefine a data member? For example, suppose we have :

```
class foo {
public:
   int a;
   char *b;
};

class bar : public foo {
public:
   float c;
   int b;
}
```

   Does the representation of a `bar` object contain one `b` or two? If two, are both accessible, or only one? Under what circumstances?

4. Why does multiple inheritance complicate the vtable implementation when using dynamic method binding? Describe one method of handling it for a class `C` that inherits from two classes `A` and `B`. What would the resulting assembly pseudo-code look like (assume a RISC-style assembly-level architecture) to access the nth virtual method of B?