11 More Classes and Objects

Kevin Schmidt, Susan Lindsey, Charlie Dey

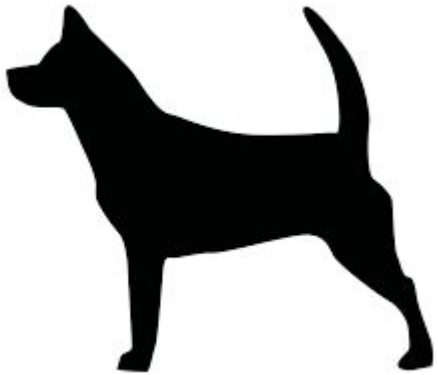
Quick Review

- An Object encapsulates both data (members) and functions on that data (methods).
- Methods are functions built into the class.
- Data member and method access can be **`private`**, **`public`** or **`protected`**.
- Methods can operate on all data members, (including private). Conversely, data members, even private, are global to the methods.

You can feed the cat. But you can't directly change how hungry the cat is, and you can't make the kitty meow, unless it's through one of the defined methods.

Bad Encapsulation

```
dog.meow()  ??
```



Methods that alter the object

```
class Vector {
    /* ... */
    void scaleby( double a ) {
    vx *= a; vy *= a; };
    /* ... */
};
    /* ... */
    Vector p1(1.,2.);
    cout << "p1 has length "
         << p1.length() << endl;
    p1.scaleby(2.);
    cout << "p1 has length "
         << p1.length() << endl;
```

OUTPUT:

```
p1 has length 2.23607
p1 has length 4.47214
```

Methods that create a new object

```
class Vector {
    /* ... */
    Vector scale( double a ) {
        return Vector( vx*a, vy*a );
    };
    /* ... */
};
/* ... */
cout << "p1 has length "
     << p1.length() << endl;
Vector p2 = p1.scale(2.);
cout << "p2 has length "
     << p2.length() << endl;
```

OUTPUT:

```
p1 has length 2.23607
p2 has length 4.47214
```

Multiple Constructors

```
Vector v1(1.,2.), v2;
cout << "v1 has length " << v1.length() << endl;
v2 = v1.scale(2.);
cout << "v2 has length " << v2.length() << endl; <--BOOO!
```

The problem is with **v2**. How is it created? We need to define two constructors:

```
class Vector {
    /*...*/
    Vector() {};
    Vector( double x,double y ) {vx = x; vy = y; };
}
```

Destructors

- Every class **`myclass`** has a *destructor* **`~myclass`** defined by default.
- The default destructor does nothing:

```
~myclass() {};
```

- A destructor is called when the object goes out of scope. Think of it as 'clean-up'.
- Great way to prevent memory leaks: dynamic data can be released in the destructor. Also: closing files.

Exercise 1 - Review Solution

Make class **`Point`** with this constructor:

```
Point(float xcoordinate, float ycoordinate);
```

Write the following methods:

- **`distance_to_origin`** returns a float.
- **`printout`** uses cout to display the point.
- **`angle`** computes the angle of **`vector(x,y)`** with the x-axis.

See solution in:

`/home/kschmidt/LectureExercises/10_Objects/ex01.cpp`

Exercise 2

Extend the **`Point`** class of the previous exercise with a method: **`distance`** that computes the distance between this point and another:

if **`p`**,**`q`** are **`Point`** objects, then

```
p.distance(q)
```

computes the distance between them.

Hint: remember the 'dot' notation for members.