• **Pointer to Class**

**Example:**

```cpp
class Car{
public:
    int x
    void drive(...){...}
};
```

```cpp
Car *p = new Car;
p->x = 5;
p->drive(...);
```

• **Classes with Pointers**

**Example:**

```cpp
class Person {
public:
    int *p;
    Person(int n){
        p = new int[n];
    }
};
```

```cpp
void foo()
{
    ...  
    Person jessica(4);
    ...
    what happens here?
}
```

```cpp
-- the this pointer
```

**Example:**

```cpp
class IceCream{ 
public
    int x;
    void foo(IceCream* cont this, int n){
        x = n;
    }
```

```cpp
the this pointer points to the calling object or a member function
```
this \rightarrow x = n;
}
;

IceCream bob, alice, tim;
bob.foo(5);

• the "Big-3": Destructor, Copy Constructor, Operator =

• Destructor: called automatically when an object ends its life; goes out of scope {}

• Copy Constructor: called when an object is declared and initialized by other object

    MyClass a = b;
    MyClass a(b);

• when passing and returning objects by value.

    foo(MyClass c)

- operator equals

    bob = tim.
    bob = tim = tim = yes;

Example:

class Dog{
public:
    int *p;
    Dog(){
        p = new int[3];
    }
    ~Dog() {} //destructor
    Dog( const Dog &rhs ) : p(rhs.p) {} //copy-cons
    const Dog& operator=( const Dog &rhs ) // operator=
}
Example:

```cpp
class Dog{
public:
    int *p;
    Dog(){
        p = new int[3];
    }
    ~Dog() { //destructor
        delete [] p;
    }
    Dog( const Dog &rhs ) { //copy-cons
        p = new int[3];
        *this = rhs;
    }
    const Dog& operator=( const Dog &rhs )// operator=
    {
        if( this != &rhs ){ //alias test
            for(int k=0; k<3; k++)
                p[k] = rhs.p[k];
        }
        return *this;
    }
};
```

```cpp
int foo(){
    ... ...
    Dog jeff;
    Dog fido;
    Dog snu = jeff;
    fido = jeff;
}
```

Whenever you have a class with pointer member, always consider overwriting the Big-3.
Stay tuned to e-mails, HW #1 to posted soon.

= Dynamic Arrays

int **p;
double ***r;