Making your own datatypes: classes

```
class G {
  public:
    // what is a G made of?
    int a; // member variables
    int b;
    void print(); // member function
};

void G::print() {
    cout << "a: " << a << "\n";
    cout << "b: " << b << "\n";
}
// how to use G?
int main() {
    G x; // x is of type G.
    x.a = 7;
    x.b = 99;
    x.print();
      :
```

---

Constructors

- Called automatically upon creation of variable.

```
class G {
  public:
  G();                  // Note: no return type,
  G(int ia, int ib);    // name ≡ name of class
```

```
    :
};
G::G() { a=0; b=0; }
G::G(int ia, int ib) { a=ia; b=ib; }
                    // (*this).a = ia;
                    // this->b = ib;
// (*this) is "the variable whose
              member function is being
               called"
```

---

```
G::G() {
    int a;
    a = 0;
      //
  this->a = 0;
// datatype of "this" =? G*
```