

# **Operator Overloading** · Earlier in the semester we created a class for rational numbers · An example of how a client would use that class is: Rational cRat1(3, 4); Rational cRat2(2, 5); Rational cRat3, cRat4; cRat3 = cRat1.multiplication(cRat2); cRat4 = cRat1.addition(cRat2); It would be much easier if we could instead write

cRat3 = cRat1 \* cRat2; cRat4 = cRat1 + cRat2; CS250 Introduction to Computer Science II

# **Operator Overloading**

4/6/09

4/6/09

- · We defined a print function to output the contents of a set cRat1.printRational();
- Wouldn't it be more efficient and more consistent with C++ if we could write

CS250 Introduction to Computer Science II

cout << cRat1;</pre>

### The How of Operator Overloading

- Write a function definition for the operator, but the function name becomes **operator** followed by the symbol
  - $\circ$  operator<<
  - $\circ$  operator+

4/6/09

4/6/09

4/6/09

o operator==

### **Operator Overloading**

 Operator overloading can be achieved in one of two ways

CS250 Introduction to Computer Science II

- A member function of the class
- A friend function of the class
- Using operator overloading through member functions has the restriction that the object of the class must always be to the left of the operator

CS250 Introduction to Computer Science II

Not useful for the insertion operator <</li>

### friend Functions

- Only the member functions of a class have direct access to the private data members of the class
- friend functions are friends of the class that are defined outside of the class but still have access to private data members

CS250 Introduction to Computer Science II

### friend Functions

- The function prototype is placed in the class, preceded by the keyword friend
- The function is able to directly access the private data members

## friend Functions

4/6/09

• The friend function could be a member function in another class

CS250 Introduction to Computer Science II

- A class could also be made a friend of an existing class
  - In this case, every member function of the friend class will have access to this class's private data

CS250 Introduction to Computer Science II

### operator<<

4/6/09

4/6/09

- << must be overloaded using friend functions
- The return value of operator<< is an ostream&
- The arguments will be the output stream and an object of the class

CS250 Introduction to Computer Science II

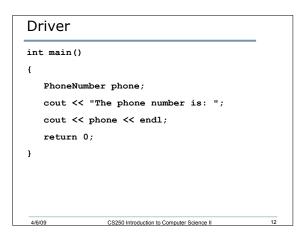
```
Example
```

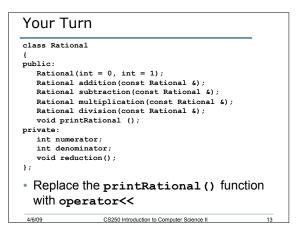
4/6/09

CS250 Introduction to Computer Science II

10







### **Overloading Binary Operators**

- Examples of binary operators that can be overloaded are +, -, \*, and /
- Unlike the insertion and extraction operators that are overloaded as friend functions, the binary operators are overloaded as regular member functions of the class

CS250 Introduction to Computer Science II

14

15

### Example

4/6/09

4/6/09

```
    Let us add functionality to the Rational class
to support the following:
Rational cRat1(3, 4);
Rational cRat2(2, 9);
Rational cRat3;
cRat3 = cRat1 + cRat2;
```

CS250 Introduction to Computer Science II

# Member Function Prototype

 In the class interface, let us add the function prototype for the overloaded operator

CS250 Introduction to Computer Science II

RationalNumber operator+(const

RationalNumber &);

16

### Member Function Definition

4/6/09

4/6/09

RationalNumber operator+(const RationalNumber & r) { RationalNumber add; add.numerator = numerator \* r.denominator + denominator \* r.nominator; add.denominator = denominator \* r.denominator; add.reduction(); return add; } 4009 CS250 Introduction to Computer Science II 17

# Your Turn • Overload the multiplication operator in the rational class

CS250 Introduction to Computer Science II

18