

---

## Operator Overloading

Section 11.6

3/9/07

CS250 Introduction to Computer Science II

1

---

---

---

---

---

---

---

---

## Operator Overloading

---

- A couple of weeks ago 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;
cRat13 = cRat1.multiplication(cRat2);
cRat4 = cRat1.addition(cRat2);
```
- It would be much easier if we could instead write

```
cRat3 = cRat1 * cRat2;
cRat4 = cRat1 + cRat2;
```

3/9/07

CS250 Introduction to Computer Science II

2

---

---

---

---

---

---

---

---

## Operator Overloading

---

- 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

```
cout << cRat1;
```

3/9/07

CS250 Introduction to Computer Science II

3

---

---

---

---

---

---

---

---

## The How of Operator Overloading

- Write a function definition for the operator, but the function name becomes **operator** followed by the symbol
  - **operator<<**
  - **operator+**
  - **operator==**
- Two operators are used without overloading
  - **&** the address operator
  - **=** memberwise assignment

3/9/07

CS250 Introduction to Computer Science II

4

---

---

---

---

---

---

---

---

## Operator Overloading

- Operator overloading can be achieved in one of two ways
  - 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
  - Not useful for the insertion operator **<<**

3/9/07

CS250 Introduction to Computer Science II

5

---

---

---

---

---

---

---

---

## **operator<<**

- **<<** 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

3/9/07

CS250 Introduction to Computer Science II

6

---

---

---

---

---

---

---

---

## Example

```
class PhoneNumber
{
    friend ostream &operator<<(ostream&, const
    PhoneNumber &);
    friend istream &operator>>(istream&, PhoneNumber &);
private:
    char areaCode[4]; // 3-digit area code and null
    char exchange[4]; // 3-digit exchange and null
    char line[5];     // 4-digit line and null
};
```

3/9/07

CS250 Introduction to Computer Science II

7

---

---

---

---

---

---

---

---

## Definition

```
ostream &operator<<(ostream &output, const
    PhoneNumber &num)
{
    output << "(" << num.areaCode << ") "
           << num.exchange << "-" << num.line;
    return output;
}
```

3/9/07

CS250 Introduction to Computer Science II

8

---

---

---

---

---

---

---

---

## Driver

```
int main()
{
    PhoneNumber phone;
    cout << "The phone number is: ";
    cout << phone << endl;
    return 0;
}
```

3/9/07

CS250 Introduction to Computer Science II

9

---

---

---

---

---

---

---

---

## Your Turn

```
class Rational
{
public:
    Rational(int = 0, int = 1);
    Rational addition(const Rational &);
    Rational subtraction(const Rational &);
    Rational multiplication(const Rational &);
    Rational division(const Rational &);
    void printRational ();
private:
    int numerator;
    int denominator;
    void reduction();
};
```

- Replace the `printRational()` function with `operator<<`

3/9/07

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

3/9/07

CS250 Introduction to Computer Science II

11

---

---

---

---

---

---

---

---

---

---

## Example

- 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;
```

3/9/07

CS250 Introduction to Computer Science II

12

---

---

---

---

---

---

---

---

---

---

## Member Function Prototype

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

```
RationalNumber operator+(const  
                        RationalNumber & r);
```

3/9/07

CS250 Introduction to Computer Science II

13

---

---

---

---

---

---

---

---

## Member Function Definition

```
RationalNumber operator+(const  
                        RationalNumber & r)  
{  
    RationalNumber add;  
    add.numerator = numerator * r.denominator +  
                    denominator * r.nominator;  
    add.denominator = denominator * r.denominator;  
    add.reduction();  
    return add;  
}
```

3/9/07

CS250 Introduction to Computer Science II

14

---

---

---

---

---

---

---

---

## Your Turn

- Overload the multiplication operator in the rational class

3/9/07

CS250 Introduction to Computer Science II

15

---

---

---

---

---

---

---

---