IIT Bombay

Computer Programming

Dr. Deepak B Phatak
Dr. Supratik Chakraborty
Department of Computer Science and Engineering
IIT Bombay

Session: Operator Overloading

Quick Recap of Relevant Topics

IIT Bombay

- Object-oriented programming with structures and classes
- Accessing data members and member functions
- Constructors and destructors
- Function calls with structures and classes

Overview of This Lecture

- Customizing operators for classes
  - Operator overloading
  - Assignment overloading

Acknowledgment

- Much of this lecture is motivated by the treatment in

  An Introduction to Programming Through C++

  by Abhiram G. Ranade

  McGraw Hill Education 2014

- Examples taken from this book are indicated in slides by the citation **AGRBook**

Motivating Operator Overloading

IIT Bombay

Recall Class V3

```
class V3 {
    private:
        double x, y, z;
    public:
     … Constructor,  destructor, other member functions …
     V3 sum (const V3 &b) {
        V3 v;
         v.x = x + b.x; v.y = y + b.y; v.z = z + b.z;  return v;
      }
};
```

Motivating Operator Overloading

IIT Bombay

Recall Motion Simulator

```
int main() {
   V3 vel, acc, pos;
   V3 currDispl, currPos;
   double t, deltaT, totalT;
      … Some code here …
    while (t <= totalT) {
       currDispl = (vel.scale(t)).sum(acc.scale(0.5*t*t));
       currPos = currDispl.sum(pos);
       t = t + deltaT;
    }
      … Some code here …
}
```

Isn’t that too clumsy?

Motivating Operator Overloading

IIT Bombay

```
int main() {
   V3 vel, acc, pos;
   V3 currDispl, currPos;
   double t, deltaT, totalT;
      … Some code here …
    while (t <= totalT) {
      currDispl =  (vel * t) +  0.5 * (acc * (t*t));
      currPos = currDispl + pos;
      t = t + deltaT;
    }
      … Some code here …
}
```

Can we write this instead?

Motivating Operator Overloading

- Normally + and * operators in C++ don't operate on V3 objects as operands
- Can we "overload" their meaning to operate on V3 objects?

Yes, indeed! C++ provides a way of achieving this!!!

Understanding Infix Operators in C++

IIT Bombay

Suppose @ is an infix operator (e.g. +, -, /, %, …)

Written between operands, as in X @ Y

In C++, the expression **X @ Y** is equivalent to $\underbrace{\textbf{X . operator@ (Y)}}$

Call to member function "operator@" of class of X
Invoked on receiver object X
Parameter passed is object Y

C++ keyword

Defining Custom Operators for Class V3

IIT Bombay

```
class V3 {
   private: double x, y, z;
   public:
     … Constructor, destructor, other member functions …
     V3 operator+ (const V3 &b) {
       return V3(x + b.x, y + b.y, z + b.z);
     }
     V3 operator* (const double factor) {
       return V3(x*factor, y*factor, z*factor);
      }
 };
```

Replaced "sum" with "operator+"

Replaced "scale" with "operator*"

Defining Custom Operators for Class V3

IIT Bombay

```
class V3 {
  private: double x, y, z;
  public:
   … Constructor, destructor, other member functions
   V3 operator+ (const V3 &b)  const  {
     return V3(x + b.x, y + b.y, z + b.z);
   }
   V3 operator* (const double factor)  const  {
     return V3(x*factor, y*factor, z*factor);
    }
 };
```

Preferable to use const. Denotes that member function cannot change receiver object

C++ Program With Overloaded Operators

IIT Bombay

```
int main() {
   V3 vel, acc, pos;
   V3 currDispl, currPos;
   double t, deltaT, totalT;
      … Some code here …
    while (t <= totalT) {
      currDispl =  (vel * t) +  0.5 * (acc * (t*t));
      currPos = currDispl + pos;
      t = t + deltaT;
    }
      … Some code here …
}
```

Invoking member function operator*

This appears problematic!
Recall: X@Y and X.operator@(Y)

Another Overloading Technique

IIT Bombay

- C++ also allows us to define operator@ as an ordinary (non-member) function, and use @ as an infix operator in expressions

```
V3 operator* (const double factor,  const V3 &b) {
      return (b * factor);
}
```

Another Overloading Technique

IIT Bombay

- C++ also allows us to define operator@ as an ordinary (non-member) function, and use @ as an infix operator in expressions

Note the order of typed operands.
Allows (factor * b) to be evaluated

```
V3 operator* (const double factor,  const V3 &b) {
      return (b * factor);
}
```

Invoking member function.
Equivalent to   b.operator*(factor)

C++ Program With Overloaded Operators

IIT Bombay

```
int main() {
  V3 vel, acc, pos;
  V3 currDispl, currPos;
  double t, deltaT, totalT;
     … Some code here …
   while (t <= totalT) {
     currDispl =  (vel * t) +  0.5 * (acc * (t*t));
     currPos = currDispl + pos;
     t = t + deltaT;
   }
     … Some code here …
}
```

Invoking member function operator*

Invoking non-member function operator*

Operators That Can Be Overloaded

IIT Bombay

- Almost all operators that you care about

Binary: + - * / % ^ & | < > == != <= >= << >> && ||

= += -= *= /= %= ^= &= |= <<= >>= []

Note the assignment operators

Unary: + - * & ! ~ ++ --

Assignment Operator

- Unlike several other operators, the assignment operator (=) is defined for all classes/structures

```
V3 a(1.0, 2.0. 3.0);
V3 b;
b = a;
```

Copy values of all data members of a to corresponding data members of b

Assignment Overloading

- We can re-define the assignment operator for a class/struct by defining the member function **operator=**

(lhs = rhs) as an assignment expression

is equivalent to

lhs.operator=(rhs)

- Definition of member function **operator=** similar to copy constructor, except that **operator=** must also return a value (like all assignment expressions)

Assignment Overloading Example [Ref AGRBook]

IIT Bombay

```
class  Queue{ private: int front, nWaiting, elements[100];
      public:
        Queue & operator=(const Queue &rhs)  {
               front = rhs.front; nWaiting = rhs.nWaiting;
               for (int i = front, j = 0; j < nWaiting; j++) {
                  elements[i] = rhs.elements[i];   i = (i + 1) % 100;
               }
             return *this;
        }
     … Other member functior
};
```

Inside a member function, "this" denotes a pointer to the receiver object

Summary

IIT Bombay

- Operator overloading in C++ as a programming convenience
- Assignment overloading as a special case of operator overloading