OVERLOADED OPERATORS AND `friend` FUNCTIONS

Reusing operators

Delroy A. Brinkerhoff

OPERATORS

- An operator is a function with a special calling syntax
- “Regular” functions:
  - `y = sqrt(x);`
  - `p = pow(b, e);`
- Operators
  - `z = x + y;`
  - `-n;`

OPERATOR OVERLOADING

- Is a form of function overloading (i.e., they are functions named `operator`☺ where ☺ is an overloadable operator)
- Cannot change the number of operands (or arguments)
- Cannot alter the precedence or associativity of an operator
- Does not change the meaning of any operator for an fundamental data type
- Cannot create a new operator (e.g., **)
- Overloaded operators should be used intuitively (e.g., in a way similar to the original meaning)

`friend` FUNCTIONS

- `friend` functions are not members of a class, but are still allowed access to private class features
- A function may be a `friend` of more than one class (called a bridge function)
- A function must be declared as a `friend` in a class
  - Can be inline
  - Can be defined outside of a class
- `friend` functions are often used with overloaded operators

OPERANDS AND ARGUMENTS

Implementation	Operator	
	Unary	Binary
Member	x.operator()	x.operator(y)
	-x	x - y
friend	operator(x)	operator(x,y)
	-x	x - y