Operators

Inst. Nguyễn Minh Huy

Contents

- Operator function.
- Special operators.
- Friend function.

Contents

- **Operator function.**
- Special operators.
- Friend function.

Operator function

- Operator on int vs. Fraction:

```
// Using int type
int main()
{
        int   a, b;
        int   c = a + b;
}
```

```
// Using Fraction type
int main()
{
        Fraction   p1, p2;
        Fraction   p3 = p1.add(p2);
}
```

```
template <class T>
T  findMin(T  a, T  b)
{
        return (a < b) ? a : b;
}
```

```
int main()
{
        int   a, b;
        int c = findMin(a, b);

        Fraction p1, p2;
        Fraction p3 = findMin(p1, p2);
}
```

Inequality between built-in and user-defined types

Operator function

- Operator function:
  - Concepts:
    - A special function.
    - Name is math symbol.
    - Syntax: ***operator*** **<math symbol>**.

      Fraction ***operator +****(Fraction p1, Fraction p2);*
  - Usage:
    - Math operator can be used on user-defined type.

      Fraction p3 = p1 **+** *p2;*
    - Can be overloaded.

      float ***opeartor +****(Fraction p, float num);*
      float x = p1 **+** *3.14;*

Operator function

- Classification:
  - Independent operator:

    Fraction ***operator +****(Fraction p1, Fraction p2);*

    - Does not belong to any class.
    - Number of arguments = operator n-nary.
  - Class operator:

    Fraction ***Fraction::operator +****(Fraction p);*

    - A method of class.
    - Number of arguments = operator n-nary - 1.
  - **They act the same!!**

Operator function

- Re-definable operators:

N-nary	Group	Operator
Unary	Inc / Dec	++, --
	Math sign	+, -
	Bit	!, ~
	Pointer	*, &
	Type-cast	int, float, double, …
Binary	Arithmetic	+, -, *, /, %
	Comparison	>, <, ==, >=, <=, !=
	Logic	&&, ||, &, |
	Input / Output	<<, >>
	Assignment	=, +=, -=, *=, /=, %=
	Array indexing	[ ]

Operator function

- Re-defined operator limitations:
  - Cannot create new operator.
  - Cannot re-define operator on built-in types.
  - Cannot change operator n-nary.
  - Cannot change operator priority.

Operator function

- Dr. Guru advises:
  - Rule of re-defining operator:
    - Name: **operator <math symbol>**.
    - Arguments: **n-nary and operands.**
    - Return type: **operator result**.
- Practice:
  - Operator > (class Fraction).
  - Operator [ ] (class Array).

Contents

- Operator function.
- **Special operators.**
- Friend function.

Special operators

- Assignments (=, +=, -=, *=, /=, …):
  - Provide operator **+=** for **Fraction**.
  - Operator n-nary?
  - Return result?

```
Fraction&  Fraction::operator +=( const Fraction &p );
```

Special operators

- Increase/decrease: (++, --):
  - Provide operator **++** for **Fraction**.
  - Operator n-nary?
  - Return result?
  - Prefix vs. postfix?

```
Fraction&  Fraction::operator ++( );            // Prefix.
Fraction  Fraction::operator ++( int x );       // Postfix, fake argument.
```

Contents

- Operator function.
- Special operators.
- **Friend function.**

Friend function

- Independent operator:
  - Provide operator **+** for **Fraction**.
  - Use independent operator.

```
Fraction  operator + ( const Fraction &p1, const Fraction &p2 );
```

  - How to access ***private***?
- Operator <<:
  - Provide operator **<<** for **Fraction**.

```
Fraction  p( 1, 3 );
std::cout  <<  p;
```

  - Which class operator **<<** belong to?

Friend function

- Friend function:
  - Function can access class **private** members.
  - Usage:
    - Declaration: ***friend*** **<function prototype>**, inside class.
    - Implementation: like an independent function, outside class.

```
class Fraction
{
        friend std::ostream& operator <<( std::ostream &os, const Fraction &p);
};
std::ostream & operator <<( std::ostream &os, const Fraction &p)
{
        os << p.m_num << "/" << p.m_den << endl;
        return os;
}
```

Summary

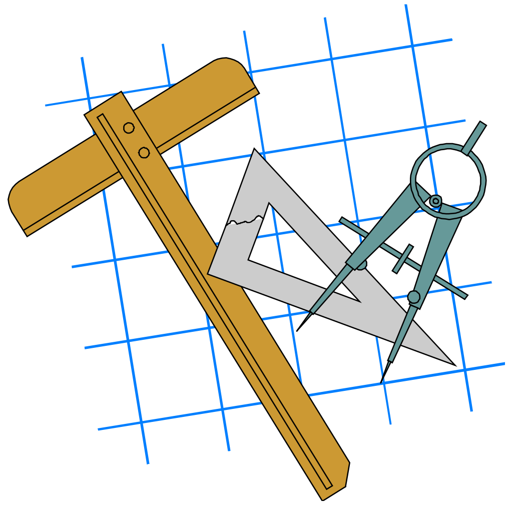
- Operator function:
  - Function having math symbol as name .
  - Provide operators for user-defined type.
  - Classification:
    - Independent operator.
    - Class operator.
- Special operators:
  - =, +=, -=, ++, --.
- Friend function:
  - Function can access private members.

Practice

- Practice 4.1:

  Provide class **Fraction** the following operators:

  - Arithmetic: +, *.
  - Comparison: >, <, ==, >=, <=, !=.
  - Assignment: =, +=, *=.
  - Inc/Dec: ++, -- (add/subtract 1 unit).
  - Type-cast: (float), (int).
  - Input/Output: >>, <<.

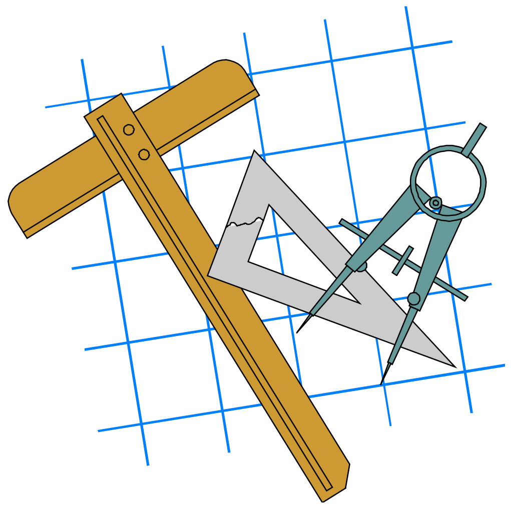


Practice

- Practice 4.2:

  Provide class **Monomial** the following operators:

  - Arithmetic: + (same exponent), *.
  - Comparison: >, <, ==, >=, <=, !=.
  - Assignment: =, += (same exponent), *=.
  - Inc/Dec:
    - ++, -- (add/subtract exponent).
    - ! (derive).
  - Input/Output: >>, <<.

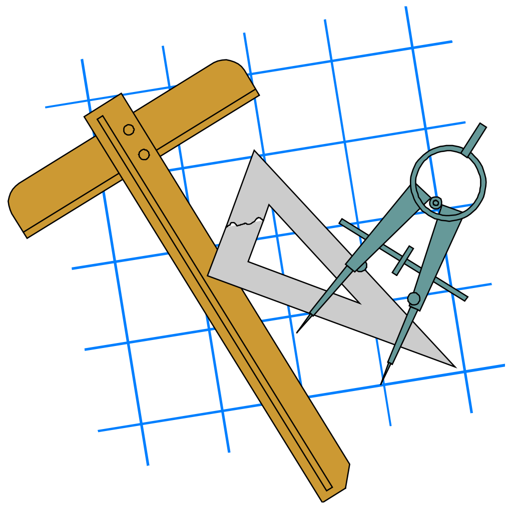


Practice

- Practice 4.3:

  Provide class **Array** (elements of any type) the following operators:

  - Assignment: =.
  - Array indexer: [ ].
  - Type-cast: (T *) (to T pointer).
  - Input/Output: >>, <<.



Practice

- Practice 4.4:

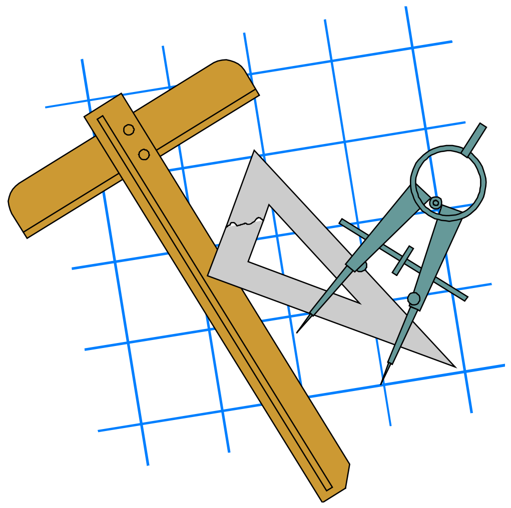
  Construct class **Time** having the following methods:

  (Constructors)

  - Initialize default time with current time.
  - Initialize time from hour, minute, second.
  - Initialize time from absolute seconds (within a day).
  - Initialize time from another time object.

  (Getters/Setters)

  - Get/Set hour, minute, second.
  - Get/Set absolute seconds (within a day).



Practice

- Practice 4.4:

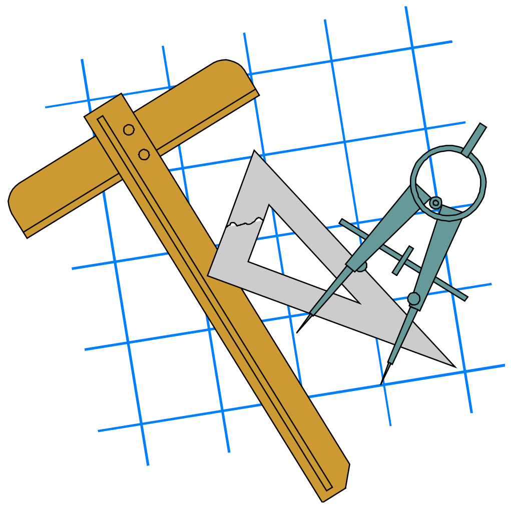
  Construct class **Time** (continue):

  (Process)

  - Compare to another time object.
  - Calculate distance (in seconds) to another time object.
  - Add seconds.

  (Operators)

  - Comparison: >, <, ==, >=, <=, !=.
  - Arithmetic: + (add seconds).
  - Inc 1 second: ++.
  - Input/Output: >>, <<.



Practice

- Practice 4.5:

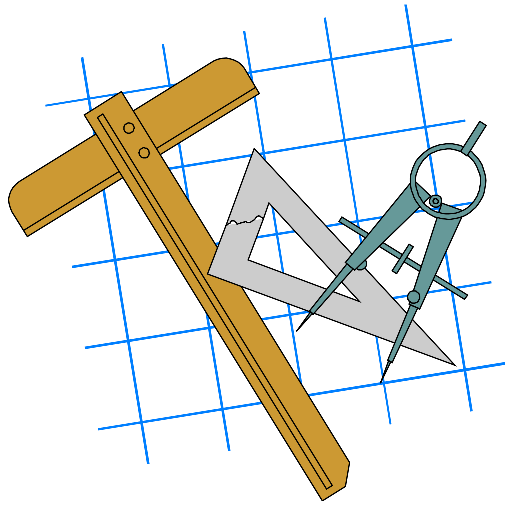
  Construct class **Date** having the following methods:

  (Constructors)

  - Initialize default date with current date.
  - Initialize date from day, month, year.
  - Initialize date from year, absolute days (within a year).
  - Initialize date from another date object.

  (Getters/Setters)

  - Get/Set day, month, year.
  - Get/Set absolute days (within a year).
  - Get day of week.
  - Get week of year.



Practice

- Practice 4.5:

  Construct class **Date** (continue):

  (Process)

  - Check leap year.
  - Compare to another date object.
  - Calculate distance (in days) to another date object.
  - Add days.

  (Operators)

  - Comparison: >, <, ==, >=, <=, !=.
  - Arithmetic: + (days).
  - Inc 1 day: ++.
  - Input/Output: >>, <<.