5 friend or member?

- In general, use a **member function** if the task performed by the function **involves only one object**
- In general, use a **non-member function** if the task performed by the function **involves more than one object**
  - Choosing to make **the non-member function a friend** is a decision of efficiency and personal taste
- For example, operator += involves two objects.
  - `time1 += time2`
    - Both **time1** and **time2** objects go to the function input arguments.
- But it is possible to be implemented as a member function.

5.1 operator += as a member

- += could be declare this way for the **TimeOfDay** class

```
TimeOfDay operator += (const TimeOfDay& t2);
// precondition : t1 and t2 have values.
// comput t1 + t2 and store result to t1
// returns t1
```

- +=could be defined this way for the **TimeOfDay** class

```
TimeOfDay TimeOfDay::operator += (const TimeOfDay& t2)
{
    hours += t2.hours;
    minutes += t2.minutes;
    checktime();
    return *this;
}
```

  - This is a member function of object **time1**, therefore, this can access to the member variables without object name.
  - Also this can access to the member variables of other object of same class (likewise friend functions).
  - operator += has to return the first object, **time1**.
    - But how do we access to the object in a member function of itself?
    - Pointer variable **this** is it!
    - **this** is a predetermined pointer variable in any class.

See **sample36.cpp**