2020 年 OOP 上机实验(5)

(Ver. 0.68 2020/04/28 wangxp@fudan.edu.cn)

一、实验目标

1） 编写完整的类，熟悉类的构造函数、析构函数、拷贝构造函数；熟悉操作符重载（含输入输出操作符）；熟悉成员函数与非成员函数的区别；熟悉友员函数；熟悉类的静态属性；

2）理解完整的类代码包含成员属性、成员函数、静态属性、静态成员函数；以及非成员函数，特别是输入输出重载操作符，不能作为成员的两元或多元操作数和关系操作符；

3） 理解基于测试的程序开发过程；

二、实验内容

ISO 是一个标准化组织,致力于多个方面的标准开发。本次实验的内容主要是练习 ISO 制定的标准 week date[1]，下面的代码已经处理并进行了规范。

从标准中，我们抽取了 year 这个类，其接口定义在 year.h 文件中。本次实验提供的文件包括：year.h,year.cpp,test_year.cpp,其中 year.h 和 test_year.cpp 无需修改，year.cpp 需要补充，**最后仅需提交 year.cpp：需要补充的函数在//TODO 标记下**。

关于 year.h 文件，有如下几点需要说明：

1)year 类是一个纯数据类，不包含资源，因此无需定义拷贝构造函数、拷贝赋值操作符、移动构造函数、移动赋值操作符、析构函数；相反，只定义了默认的构造函数，以及接受一个 int 值的构造函数；

2)year 类定义了两个静态函数 min/max，限制所表示的年分必须在 min 和 max 的范围之间，**按照 C++的惯例表示的范围为[min, max)， 即左闭右开区间**；

3)year 类定义了一元操作符+/-；二元操作符+=/-=；自增(前缀、后缀)操作符；自减（前缀、后缀）；--这些操作符统统定义为成员函数；

4)6 个关系操作符，两元+/-操作符(重载了 year/years 类型)，输出操作符<<，以及字符量(literals)的重载操作符[2]

```
inline namespace literals
{
    year operator"" _y(unsigned long long y);
}
```

;--这些操作符定义为非成员函数。

关于 test_year.cpp 文件，有如下说明：

1)关于 assert 和 try_except 处理。我们建议大家在代码中大量使用 assert，而不是使用 try_except 结构；使用 assert 时，确保包含了标准头文件<cassert>;

2)在 test_year.cpp 中，针对 year 类的构建、比较操作、算术运算等操作符进行了确认。如果你写的代码一切 OK，那么将 year.cpp 和 test_year.cpp 编译出来的可执行文件，在运行时将不会有任何输出——对于测试而言，**没有消息就是最好的消息**——表示你写的代码通过了测试。

3） 在 test_year.cpp 中，还有一大段被注释的代码，请自行查阅 cppreference，弄清楚这些语句的含义。

参考资料

1. ISO week date。 参考 https://en.wikipedia.org/wiki/ISO_week_date
2. 自定义的字符常量。 参考 https://blog.csdn.net/K346K346/article/details/85322227